# -=-=-=-= Capítulo 1 – Aula 1 =-=-=-=-

Links úteis:

- <https://github.com/gustavoguanabara> | Repositório de códigos no GitHub

- <https://gustavoguanabara.github.io> | PDFs das aulas, exercícios e desafios

# -=-=-=-= Como surgiu a internet =-=-=-=-

Foi criada para cumprir objetivos de comunicação militar durante a Guerra Fria (1969) entre os Estados Unidos e a União Soviética. A União Soviética lançou o primeiro satélite “Sputnik”. Para igualar a competição, os Estados Unidos criaram uma empresa chamada DARPA (Defence Advanced Research Project Agency).

Tempos depois a ARPANET foi desenvolvida como uma rede de backups de dados entre bases militares com o objetivo de criar cópias dos dados armazenados caso alguma das bases fosse destruída.

Como na época cada computador tinha o seu próprio jeito de se comunicar, foi criado o protocolo NCP (Network Control Protocol) para estabelecer um meio principal de comunicação entre computadores. Porém, um grande empecilho do NCP era a sua necessidade de bloquear a comunicação dos demais dispositivos para que dois deles mantivessem uma conexão ativa. Com o passar dos tempos e o surgimento de diversos outros aparelhos que necessitavam de comunicação, surgiu o TCP (Transfer Control Protocol) criado por Bob Kahn. O TCP não tinha capacidade de identificar as diversas máquinas que tentavam se comunicar na rede, foi então que o protocolo IP (Identification Protocol) teve o seu desenvolvimento para atribuir identidades únicas para cada dispositivo. A junção deles ficou conhecida como TCP/IP.

Com o grande crescimento na demanda da rede de comunicação por militares e universidades, os militares migraram para a sua própria rede chamada MILNET. Enquanto as universidades ficaram em uma rede chamada NSFNET (National Science Foundation) além de redes comerciais. Apesar de serem redes distintas, ainda havia o interesse de que as redes distintas entre si conseguissem se comunicar (Interconnect Network) > Internetworking > Internet.

# -=-=-=-= Como é a estrutura da internet =-=-=-=-

Cliente: Computador que demanda/pede o acesso das informações

Servidor: Computador que recebe os pedidos e fornece informações armazenadas.

O protocolo TCP fragmenta partes da informação em pacotes que serão transmitidos por rotas, tempos e caminhos diferentes até o computador do Client, onde serão remontados para criar o arquivo completo. O protocolo IP é o responsável por identificar as máquinas de origem e destino deste processo.

Antigamente a internet era composta apenas de uma tela preta com informações escritas nela. Funcionava através do Protocolo Gopher sem usar o mouse, apenas o teclado.

Tempos depois, Tim Berners-Lee elaborou um protocolo que integrava links sensíveis que abriam endereços web vinculados ao texto (hyperlinks), criando assim o Protocolo HTTP (ele também criou o HTML – HyperText Markup Language e a WWW – Worl Wide Web).

Marc Andreeesen criou o primeiro navegador (Mosaic) compatível com HTTP.

Quando acessamos os sites com WWW, eles são sub-redes dentro da internet especializadas em HTTP.

# -=-=-=-= Como a internet funciona =-=-=-=-

Computadores se comunicam usando transmissão de sinais ou a ausência deles atribuídos aos números 0 e 1. São chamados de Bits. Formam portanto “ondas quadradas”.

Um conjunto de 8 bits foram um byte. É o conjunto mínimo para representar um dado. Tabela de código multibyte UTF-8.

Múltiplos do byte (exemplo de comparação, múltiplos do quilo > grama > tonelada). 8 bits = 1 byte. 1024 bytes = 1 KiloByte; 1024 KB = 1 MegaByte; 1024 MB = 1 GigaByte; 1024 GB = 1 TeraByte; 1024 TB = 1 PetaByte; 1024 PB = 1 ExaByte; 1024 EB = ZetaByte. 1024 ZB = 1 YotaByte.

# -=-=-=-= Diferença entre MB e Mb =-=-=-=-

MB = MegaBYTES (armazenamento)

Mb = MegaBITS (transmissão)

# -=-=-=-= Como nos conectamos? =-=-=-=-

Para o computador se conectar com a internet em busca das informações desejadas que estão armazenadas em servidores, ele transmite ondas quadradas para o modem/roteador que fará a conversão em ondas senoidais (modulação) para transmitir as informações adianta. O caminho contrário, ou seja, de recepção da informação externa é convertida de ondas senoidais para ondas quadradas (demodulação). A identificação das origens e servidores é feita por meio de IPs que são identidades atribuídas a cada máquina.

**DNS (Domain Name System)** – Servidores dedicados a armazenar os endereços de IPs de cada domínio/nome existente.

**Domínio**: Nome que identifica o site. Nome único. Precisa ser pago anualmente. Vários TLD (Top Level Domains).

**Hospedagem**: Local onde o site estará hospedado. Espaço para armazenar os arquivos. Pago mensalmente. Espaço, memória, recursos, etc.

**URL(Uniform Resource Locator)**: Exemplo www.github.com/gustavoguanabara

github -> Domínio

.com -> TLD (Top Level Domain) | Existem também os GTLD (Generic Level Domain. Exemplo .com .edu .gov .store) | CCTLD (Country Code. Exemplo .br .us .uk)

www -> Sub-Domínio

/gustavoguanabara -> Caminho que indica a ramificação/localização da página a ser apresentada

https://www.cursoemvideo.com/curso/html5-css3-modulo1/aulas/modulo-1-2/modulos/capitulo-3-aula-1-a-diferenca-entre-html-css-e-javascript

https://ava.ufms.br/login/index.php

https://www.youtube.com/feed/subscriptions

\*https:// -> protocolo web. Não é sub-domínio

# -=-=-=-= Diferenças entre a HTML, as CSS e JavaScript =-=-=-=-

A linguagem HTML (Hyper Text Markup Language) | Áreas de marcação (tags) que te levam para outras áreas

As folhas de estilo CSS (Cascading Style Sheets) | Uma vai modificando a característica padrão das outras em cascata

Não são linguagem de programação. Mas podem ser chamadas apenas de linguagem. “Eu desenvolvo em HTML e CSS”

* **HTML** é uma linguagem focada em conteúdos (texto, imagem, vídeo, tabelas, listas, etc)
* **CSS** tem foco em design (cor, tamanho, posicionamento, sombras, etc)
* **Java Script** aplica as interações do site (menus interativos, animações, pop-ups, validações, etc)

**Tags de HTML são construídas utilizando <>**

**<h1>** Exemplo de título **</h1>** | <h1> é a abertura de tag | “Exemplo de título” é o conteúdo | </h1> fechamento de tag

**<p>** Exemplo de parágrafo **</p>**

**<img src=”**foto.png**” alt=”**Exemplo de foto**”>** <- Existe algumas tags que não possuem fechamento | alt é o parâmetro e “Exemplo de foto é o valor do parâmetro”

**Estilos em CSS**

Feito por seletores

**H1{**

**font-family:** Arial; 🡨 Declaração

**font-size**: 20pt; 🡨 Medida em pontos. Existe também a medida px em pixels

**color**: blue; 🡨 Color é a propriedade e blue é o valor

**}**

# -=-=-=-= Estrutura básica de um documento HTML =-=-=-=-

<!DOCTYPE html> #Tag específica de configuração para informar que o programa está em HTML5

<html lang=”pt-br”> #Tags de abertura e fechamento da tag html | tag indicando que o idioma do site será em português

<head> #Configurações são realizadas no campo head

<meta charset=”UTF-8”> #Indica que o site será compatível com o padrão de linguagem UTF-8 que suporta acentos, cedilha e outras características da língua portuguesa

<meta name=”viewport”

Content=”width=device-width,initial-scale=1.0”> #Indica que o site ocupará todo o espaço disponível da tela e terá escala em 100% de zoom

<title>Document</title>

<body> #O conteúdo do site é feito no body

<h1>Olá, mundo!</h1> #Exemplo de título como conteúdo do site

</html>

# -=-=-=-= Como funciona HTML e CSS =-=-=-=-

Após informar a URL do site desejado, o servidor DNS retornará o endereço IP correspondente ao texto da URL para o servidor buscar e retornar o conteúdo desejado. Neste caso os códigos em HTML e CSS serão copiados/fornecidos ao navegador que construirá a aparência da página diretamente para o computador.

# -=-=-=-= Front-End, Back-End e Full Stack =-=-=-=-

Site Estático: O mesmo site é replicado para todas as pessoas que a requisitarem.

**Front-End**: Utiliza ferramentas HTML5, CSS e JavaScript que funcionam do lado do cliente (cliente-side) com foco na interatividade/experiência do usuário e apresentação dos recursos ao usuário.

**Back-End**: Foca em como o código interage e se comunica com o servidor (server-side). Tecnologias back-end: PHP, Java Script (para servidor), C#, Python, Ruby e Java.

**Full Stack**: Desenvolvedor que domina tanto o Front-End/Cliente Side e o Back-Ende/Server Side. Este processo para se tornar um desenvolvedor Full Stack leva muito tempo e experiência, o que é natural.

# -=-=-=-= Instalando ferramentas para desenvolvimento =-=-=-=-

* Navegador Google Chrome
* Editor de código Visual Studio Code
* Programa de manipulação de imagem gratuito Gimp

# -=-=-=-= Iniciando em HTML =-=-=-=-

Extensões instaladas:

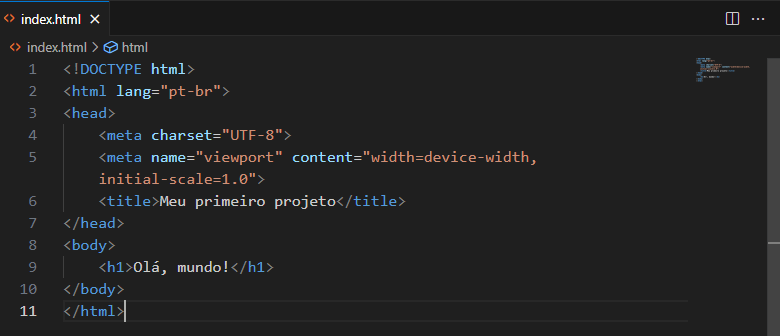
Java Script – ESLint

PHP – PHP Debug e PHP Extension Pack

Ao criar uma nova pasta que armazenará o código, é possível abrí-la direto no programa utilizando o botão direito -> abrir com Code

O HTML sempre usará um arquivo básico chamado index.htm e a sua estrutura inicial pode ser rapidamente criada escrevendo “!” e clicando na primeira opção

\* Sempre utilize letras minúsculas nos nomes dos arquivos!



Doctype define o arquivo como escrito em HTML5

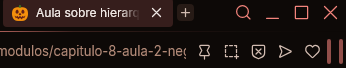
Html lang=”pt-br” define a linguagem do site como português brasileiro

Meta charset=”UTF-8” torna o site compatível com elementos da língua portuguesa (acentos, cedilha, etc)

Meta name=”viewport” contente=”width=device-width, initial-scale=1.0” determina que o site será mostrado no tamanho máximo da tela com zoom em 100%

# -=-=-=-= @ | Lista de tags em HTML=-=-=-=-

# <title> Título/Nome do site



# <link:favicon> Adicionar ícone da página do site

<link rel= “shortcut icon” href=”NomeDaImagemDoÍcone” type=”image/x-icon”>

Mais informações na categoria “Adicionando ícones de aba no site” deste documento

# [!!] Tags obsoletas aparecem em vermelho no código do VS Code

Tags como <big>, <b>, <i> e outras ainda funcionam mas não estão recebendo mais suporte e poderão parar de funcionar a qualquer momento. Estas são tags que alteram apenas a forma do conteúdo sem dar significado semântico para eles. Existem substitutos mais atualizados que estão descritos neste documento.

# <h1> Título de nível 1 / cabeçalho / header



# <p> Parágrafo

**<p>** - Parágrafo

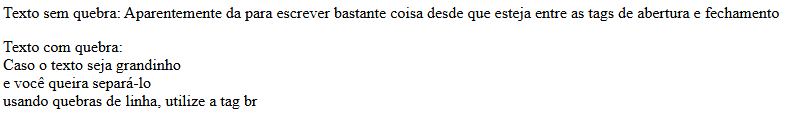


# <hr> Linha Horizontal (Horizontal Row)

(não precisa de tag de fechamento)



# <br> Quebra de Linha (Break Row)



# Escrever <> como texto e não como tag “&lt;” e “&gt;”

Escrever os símbolos <> numa página html sem o programa identificar como criação de tag, use:

**&lt; para <** (Less Than)

**&gt; para >** (Greater Than)



# <!----> ou /\* \*/ Comentar no código

**<!--Texto-->** ou **/\* Texto \*/**| Adicionar comentários no código | Escreva entre os dois hifens de cada lado



# <Strong> Escrever texto em Negrito de maneira semântica (com significado)



Jeito incorreto: utilizando a tag <b> que apenas altera a forma, sem deixar o trecho do código sinalizado com algum sentido.

# <em> Escrever texto em *Itálico* de maneira semântica (com significado)



Jeito incorreto: utilizando a tag <i> que apenas altera a forma, sem deixar o trecho do código sinalizado com algum sentido.

# <mark> Escrever texto com marcação/cor atrás dele



Da para alterar a cor adicionando o parâmetro de CCS style= “background-color: COR” para fazer esta alteração apenas na tag que recebeu este parâmetro. Para isto, o comando extra precisa vir dentro de tag mark de abertura.

Caso existam muitas tags que usarão a mesma cor, o melhor jeito de alterar a cor de marcação é indo no trecho “Head” do código, adicionando a tag “Style”, escrevendo qual é a tag que terá o estilo modificado (neste caso a tag “mark”) e colocando o parâmetro “background-color: COR” dentro das chaves. Deste jeito todas as tags mark receberão a mesma cor da mesma página.

# <del> Escrever texto ~~riscado~~ no centro



Também é chamado de texto excluído.

# <ins> Escrever texto sublinhado



Também é chamado de texto incluído.

Maneira não semântica de sublinhar texto: utilize a tag <u>.

No Visual Studio Code, ao colocar a cor no parâmetro da tag, da para clicar no quadrado da cor e modificar livremente para o tom desejado que o código da cor escolhida será automaticamente implementado.

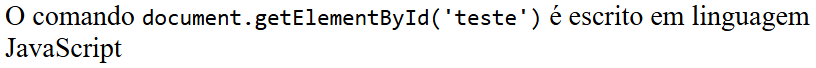
# <sup> Escrever texto sobrescrito (ex: E = m.c2)



# <sub> Escrever texto subescrito (ex: fórmula H2O)

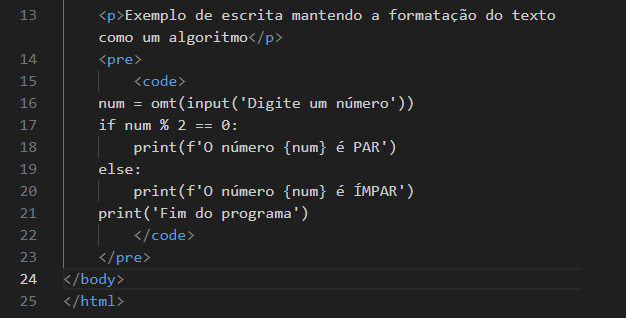


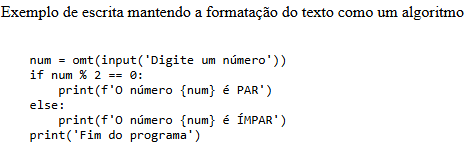
# <code> Exibir o texto com todos os caracteres possuindo o mesmo tamanho de largura (fonte mono-espaçada)



# <pre> Exibir o texto em estrutura de código na página web igual no programa onde foi escrito

Escreva o conteúdo indentado dentro da tag <pre>. Adicione também a tag <code> para formatar a exibição do seu texto mantendo todos os caracteres com o mesmo tamanho de largura.



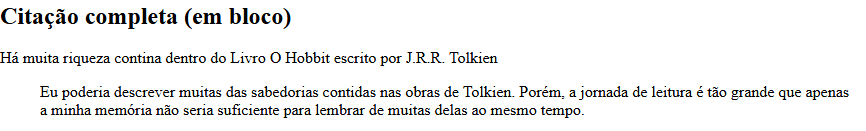


# <q> Adicionar uma citação simples entre aspas mantendo sentido semântico

Esta tag simplesmente adiciona aspas no começo e no final do texto contido nela. Porém, ela serve também para dar sentido semântico (significado) ao trecho do código que pode ser manipulado/detectado por outras funcionalidades.

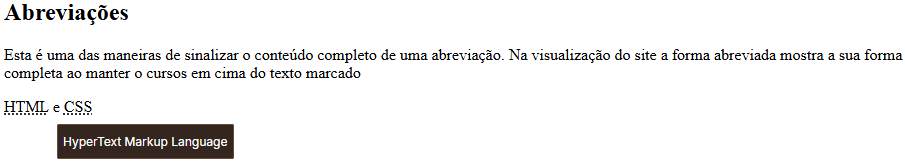
# <blockquote> Adicionar uma citação completa em bloco

Além do formato em bloco, a citação é apresentada na página com um deslocamento da margem e pode receber o parâmetro “cite=” dentro da tag de abertura para os mecanismos de busca reconhecerem o link fornecido como referência a um material externo. Visualmente a citação não mostra o link ao manter o cursos em cima do texto.

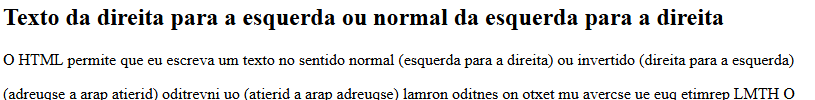


# <abbr > Sinalizando abreviações e suas formas inteiras para o site

Preencha o campo “tittle” com a forma completa da abreviação escrita. A forma abreviada fica entre a tag de abertura e a tag de fechamento abbr.



# <bdo > Escrever texto da direita para a esquerda ou o contrário

No parâmetro dir (direction) use o texto “ltr” (left to right) para exibir o texto da esquerda para direita ou “rtl” (right to left) para exibir o texto da direita para a esquerda. 

# Tabela de símbolos e seus respectivos comandos



Outros símbolos pesquisando por: miscsymbols

# Como adicionar emojis

Pesquise o site da emojipedia e procure o emoji desejado

Até da para copiar e colar o emoji no VSCode, porém, em alguns navegadores ele não funciona

O método mais seguro é procurar o código do emoji lá no final da página. Exemplo do emoji de lanterna de Halloween o código é “U+1F383”. No código usamos apenas o trecho “1F383”

Para aplicar o emoji use: **&#x1F383;**

**[\*]** Existe o emote com código “**\00A0**” que pode ser utilizado para adicionar um espaçamento junto de outro emote. Exemplo: “list-style-type: ‘\2714\00A0\00A0’ ” adiciona o emote de checkmark (código 2714) e dois espaçamentos extras. Mas cuidado, nem todo navegador tem compatibilidade para usar emotes com código.

\*Existe também o atalho “Windows + .” para buscar e adicionar um emoji rapidamente

# <img> Tag de imagem

<img **src**”” **alt** “”>

**Source** = fonte de onde a imagem está armazenada |

**Alternative** = Texto explicativo sobre a imagem. Utilizado para algoritmos e leitores de tela reconhecerem a imagem.

**[?] Como usar**: Ao salvar as imagens na mesma pasta que o código está armazenado, clique no meio das aspas Source e use o atalho Ctrl + Espaço para abrir uma lista dos arquivos disponíveis na pasta. No trecho alternative, coloque a descrição da imagem para os algoritmos reconhecerem ela.

Exemplo: <img=“logo-minecraft.png” alt=”Logo do jogo Minecraft”>

# <img> Tag de imagem com imagem em sub-pasta

Caso alguma das imagens esteja dentro de outra pasta, utilize o atalho Ctrl + Espaço, clique no ícone com o nome da sub-pasta e selecione o arquivo desejado.

Na linha do código será escrito o caminho para encontrar o arquivo e a imagem funcionará corretamente.

# <img> Tag de imagem com imagem externa (na Internet)

Copie o link/endereço da imagem usando o botão direito nela e cole o link no campo “Source” da tag img.

# <lorem> para criar trechos de textos aleatórios

Lorem ipsum dolor sit amet consectetur adipisicing elit. Facere fugit harum tempora itaque cum perferendis asperiores libero vero dolorem quos ea veritatis deserunt, sapiente assumenda corrupti nostrum dignissimos nisi! Placeat!

# <address> Sinalizar dado de endereço para interação com algoritmos ou apps

Usar a tag <address> faz com que o texto fique em itálico mas a principal função da tag é sinalizar que o elemento armazenado é um endereço que pode receber interação de aplicativos de mapeamento por exemplo.

# <ol> Ordered List (Lista Ordenada)

Entre as tags de abertura e fechamento da ordered list, você pode adicionar os itens da lista usando a tag **<li> (List Item)** em cada linha. Após adicionar um novo elemento em qualquer posição, a própria ordered list manterá a ordem correta dos itens armazenados nela.

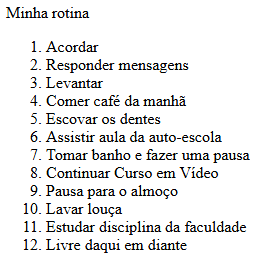
\*Da para criar listas dentro de outras listas também.

**[?]** Para alterar o tipo dos índices da lista, use o parâmetro “**type= [Opção]**” dentro da tag de abertura da Ordered List.

As opções são:

* 1 – Ordem numérica;
* A – Índice em ordem alfabética com letras maiúsculas;
* a – Índice em ordem alfabética com letras minúsculas;
* I – Índices em números romanos maiúsculos;
* i – Índices em números romanos minúsculos;

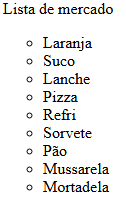
**[?]** Da também para determinar a posição de início da lista com o parâmetro “**start= [Posição]**” que seguirá a sequência normalmente à partir da posição indicada.



# <ul> Un-Ordered List (Lista Não-Ordenada)

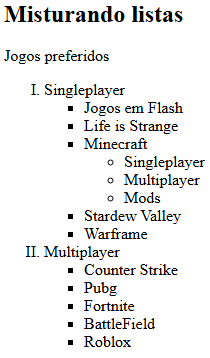
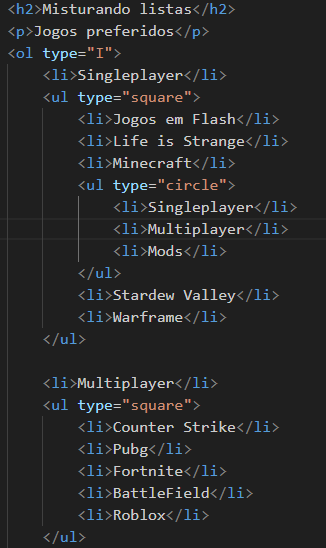
Opções de personalização dos índices da lista usando o parâmetro **type=”Tipo”**:

* Disc – Bolinha preta preenchida;
* Circle – Círculo preto sem preenchimento;
* Square – Quadrado preto sem preenchimento;



# Misturando listas

Utilizar listas ordenadas e com termos acompanhados de suas descrições fazem com que o site seja valorizado por algoritmos de busca.



# <dl> Listas de definição (Definition List)

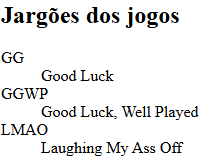
Utilizam os seguintes parâmetros para determinar a ordem/indentação dos tópicos:

# <dt> Termo de definição (Definition Term)

O termo permanece mais próximo da margem.

# <dd> Descrição de definição (Definition Description)

A descrição aparece com espaçamento maior da margem.



# <a> Tag para transformar um texto HyperLink clicável

Utilize a Tag **<a>** de Anchor (Âncora) envelopando o trecho do código que se tornará um hyperlink. No parâmetro **href”**[Link]**”** escreva o link para onde o usuário será direcionado ao clicar.

[!] Deste modo a Tag Âncora sobrescreve o site do hyperlink com a nova página visitada. Para fazer com que a página do hyperlink abra em uma nova aba, utilize os parâmetro s **target=”\_blank”** e **rel=”external”** dentro da tag de abertura da âncora.

Esta estratégia é indicada para manter a aba do nosso site ativa após o usuário querer acessar um link externo. Caso fosse um link interno, o indicado torna-se apenas sobrescrever a página antiga para evitar transtorno de abas desnecessárias.

**[\*]** Caso queira manter um link ativo mesmo sem possuir uma url, coloque # no campo “hra[link]”. Ele não redirecionará a página para lugar algum.

# <picture> Contém outras tags dentro dela para ajustar automaticamente o tamanho das imagens de acordo com o espaço da tela

Cria a tag **<img>** para aplicar a imagem desejada na tela. [!!] Esta tag img deve vir sempre abaixo das tags source que definirão o seu comportamento.

Escreva **source:media:type** no VS Code e selecione > a tag ficará escrita como **<source** 1**media=”(min-width:)”** 2**srcset=””** 3**type”image/”>**

[?] O que esse código quer dizer?

O programa vai carregar a imagem contida na tag img. Porém, se a medida da tela chegar perto do tamanho indicado em pixels, o site trocará a imagem por uma de outro tamanho mais adequado para encaixar na tela.

1 -> Parâmetro media: Mude o conteúdo de “min-width:” para “max-width:”. E na frete dos dois pontos, coloque a medida em pixels da largura da imagem desejada + 50 pixels de folga para evitar a criação de barra de rolagem antes do sistema trocar a imagem mostrada para outra de tamanho mais adequado.

2 -> Parâmetro srcset: Escreva manualmente o endereço onde está armazenada uma imagem de tamanho diferente da que está sendo utilizada na tag img. Exemplo: “imagens:fundo-medio.png”.

3 -> Descreva o tipo do arquivo. No caso de imagem: “image/png”

Após configurar este ajuste automático no tamanho da imagem, agora o passo precisa ser repetido para os ajustes de outros tamanhos que quiser utilizar.

# <audio> Padrão | Adiciona áudios para tocar no seu site sem o autoplay funcionar em todos os sites

**[!]** **Cuidado com áudios que possuam direitos autorais!**

Dica do Prof. Guanabara: Vá no seu canal do youtube > Clique no seu perfil > Youtube Studio > Biblioteca de áudio > Filtre por: Atribuição necessária (precisa publicar com o textinho da descrição) ou Atribuição Não Necessária (não precisa publicar com atribuição na descrição do áudio).

**[!!]** Fique atento sobre o tamanho do arquivo de mídia, seja ele uma imagem, áudio, vídeos, etc. Quanto melhor a qualidade ou a duração, mais pesado o arquivo. Isto causa um consumo e lentidão maior dos recursos do site e prejudica tanto o seu desempenho como o seu ranking na busca.

\*Caso tenham vários arquivos de áudio no site, o programa tentará reproduzir o primeiro e testará os demais. Da para colocar ao final uma tag **<p>** para emitir uma mensagem caso nenhum dos arquivos funcione.

Ao escolher o seu áudio, uma boa prática é renomear o arquivo para deixa-lo sem letras maiúsculas, espaços e acentos. Ajuda a evitar incompatibilidades com os programas e ferramentas.

**<áudio src=”[EndereçoDoAudio]”>** Compatível com os formatos de áudios MP3, WAV e OGG.

Adicione os parâmetros:

- autoplay (em alguns navegadores faz com que o áudio seja reproduzido automaticamente).

- controls (exibe os controles do áudio para o usuário).

- preload (pré-carrega informações do áudio). Cuidado, colocar o preload como “Auto” ou “None” faz com que o site só seja considerado 100% carregado quando todo o áudio (que pode ser um arquivo grande demais) for carregado também. A opção “Metadata” é a mais balanceada para carregar o site em troca de o áudio carregar totalmente ao apertar o play.

- loop recomeça o áudio assim que ele chegar ao fim.

# <audio> com a tag Source | Toca áudios no seu site e tem o autoplay funcionando

Crie a tag **<áudio src=””>** e apague o trecho sublinhado > Quebre a linha > Escreva entre as tags de abertura e fechamento > **“source:src”** > Selecione a opção com o mesmo nome.

No campo “src” escreve o endereço do arquivo (o atalho Ctrl + Espaço funciona).

Já no campo “type” escreva o formato de áudio correspondente com o seu arquivo (Para mp3 a extensão é “audio/mpeg” | Para ogg e wav a extensão utiliza o mesmo nome após a barra).

[?] Tenha o mesmo áudio em outros formatos de áudio para o seu site conseguir rodar outra versão dele caso alguma delas falhe.

Formatos mais utilizados e suportados: MP4, MKV, WebM, Ogv.

Programa gratuito indicado para manipular vídeos: HandBreak.

Arraste o arquivo do vídeo para o aplicativo do HandBreak > Selecione a opção “Presets/Perfil” > Fast 720p 30fps > Clique em “Encode/Iniciar” para iniciar a transformação

# <video> Exibe um vídeo em determinado formato no seu site

Utilize a tag **<vídeo src=””>**. Apague o trecho “src=” > Separe as tags de abertura e fechamento em linhas diferentes > Entre elas adicione a tag **<source:src>** para adicionar o endereço do arquivo e o tipo dele. Deste modo você pode adicionar mais de um formato do arquivo para garantir que ele será mostrado caso algum navegador demonstre incompatibilidade.

O primeiro vídeo da lista também será o primeiro a ser testado. Ao final também é possível adicionar um parágrafo para exibir um texto caso todos os vídeos falhem ao carregar.

**[!!] Cuidado:** O consumo de tráfego de rede pode ser alto com vídeos hospedados no próprio site devido à quantidade de acessos ao site e as tentativas de rodar vídeos grandes. Para isto, a melhor solução é incorporar vídeos externos.

Na tag vídeo, você pode colocar também os parâmetros:

- poster | Mostra a capa do vídeo ao carregar a página até o usuário dar play.

- width | Define uma largura máxima do vídeo e evitando do vídeo ficar imenso na tela.

- controls | Exibe os controles padrões para o vídeo naquele navegador.

-autoplay | Começa o vídeo automaticamente ao carregar o site. [!] Em alguns navegadores não funciona.

- loop | Recomeça o vídeo quando ele chegar ao fim.

**[?] Dica**: Tenha mais de uma versão do mesmo vídeo em formatos diferentes pois se algum dos formatos não for compatível com o navegador, existem outras opções a serem exibidas.

Crie mais de uma versão do vídeo em formato diferente para o site ter uma alternativa caso uma das versões não funcione em algum momento.

# <video> Incorporando vídeos externos ao site (tráfego de rede econômico)

**Acesso público (Youtube)**:

No youtube pesquise algum vídeo > Selecione o botão “compartilhar” > Escolha a opção “Incorporar” > Copie o código HTML gerado.

\*Outra vantagem do youtube é a possibilidade do cliente possuir um canal na plataforma. Deste modo o site e o canal no youtube se valorizam tanto por direcionarem tráfego entre si mas também por estarem presentes em 2 dos maiores buscadores do mundo.

**Acesso restrito (Vimeo)**: Recomendado devido a quantidade de personalizações e profissionalismo.

Caso precise regular o acesso do vídeo para apenas algumas pessoas escolhidas/assinantes, recomenda-se utilizar o Vimeo. (Qualidade boa dos vídeos porém, o carregamento pode ser um pouco mais pesado/lento).

Encontre o vídeo desejado no Vimeo > Opção “compartilhar” > Copie o código da incorporação. O botão “Opções” permite personalizar como a incorporação será exibida.

# -=-=-=-= Direitos de imagens para usar nos sites =-=-=-=-

Utilizar a aba Ferramentas da busca do google para buscar imagens com licenças mais livres.

Marcadas para reutilização com modificação – Permite uso comercial e modificação das imagens

Marcadas para reutilização – Permite uso comercial sem modificação

Marcadas para reutilização não comercial com modificação

Marcadas para reutilização não comercial – Sem modificação

# Melhores sites para encontrar imagens com licenças livres

* Unsplash
* Pexels
* Freepik
* Hallpixel
* Pixabay
* Libreshot
* WikiMedia Commons

# Muito cuidado com direito autoral e tamanho das imagens no site

[!] Ainda assim é importante verificar se realmente as imagens escolhidas estão com licenças livres

Outro detalhe bem relevante é o tamanho da imagem a ser usada no site. Quanto melhor puder ser sem prejudicar a qualidade, melhor a imagem vai carregar.

* Da para pesquisar no google por imagens com tamanho grande (boa definição) utilizando a Aba Ferramentas e buscando por imagens de tamanho grande.
* Ao pesquisar imagens filtradas em Ferramentas -> Cor -> Transparente, aparecerão apenas imagens em PNG.

- O formato PNG permite imagens com transparências.

- O formato JPEG ou JPG é o melhor formato para compactar e reduzir o tamanho das imagens. Não permite transparências.

- Extra: o formato GIF também permite transparências e animação. Porém, a qualidade é mais baixa e o tamanho é maior.

# Indicação de site para consultar os códigos de símbolos

<https://www.w3schools.com/html/html_symbols.asp>

# -=-=-=-= Manipulando imagens no Gimp =-=-=-=-

Botão direito > abrir com > Gimp > Manter formato original

Imagens com no máximo 1500 pixels de largura é o suficiente para sites. Uma outra medida que encaixa bem é 650 pixels de largura.

* **Redimensionar imagem:** Aba Imagem > Redimensionar imagem

Ao reduzir a largura, proporcionalmente deve diminuir a altura para não esticar ou comprimir a imagem. Para isto, mantenha o ícone da corrente ativo.

* **Resolução da imagem:** 50 pixels ou um pouco mais é o suficiente.
* **Exportando a imagem:** Aba Arquivo > Export as > Renomeie a imagem para indicar o tamanho que ela está > Exportar > 70% de qualidade é uma boa medida.

Após redimensionar o tamanho da imagem, o tamanho do arquivo foi bem reduzido.

# -=-=-=-= Adicionando ícones de aba no site =-=-=-=-

[\*]

- Site recomendado para pesquisar ícones: Icon Archive.

- Site para desenhar o próprio ícone: favicon.cc

- Site para transformar imagem, texto ou emji em ícones: favicon.io

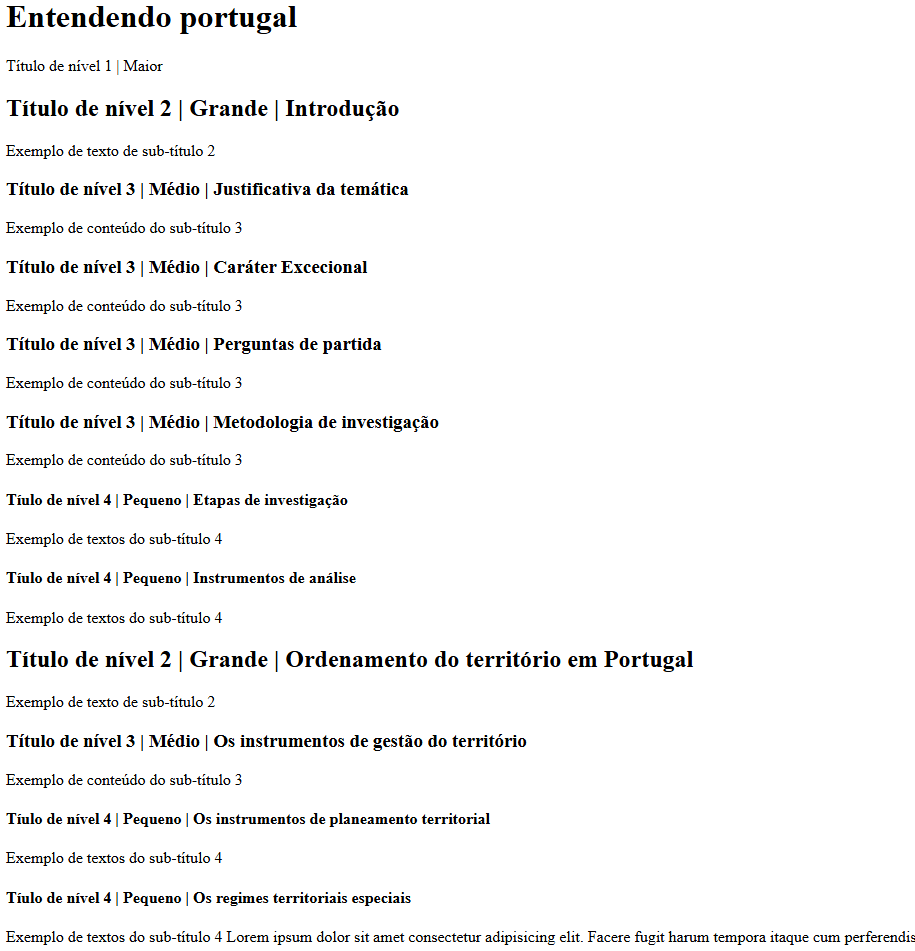
O melhor formato é ICO (de Icon) mas também serve uma imagem com poucos detalhes.

**Como usar**: Digite o seguinte código dentro do bloco “Head”-> Use a tag “Link” e selecione a opção “Link:Favicon”. Apague o que já está escrito no campo “href”, use o atalho Ctrl + Espaço e selecione o arquivo do ícone desejado

# -=-=-=-= Hierarquia de títulos =-=-=-=-

Define a ordem e organização dos títulos no site. A tag <h1> vai até h6.

Exemplo de índice



# -=-=-=-= Semântica (significado) no HTML5 =-=-=-=-

Anteriormente no HTML4 usava-se tags de estilo dentro do código HTML para alterar a sua forma.

Agora no HTML5, a linguagem está focada apenas na semântica e não na forma (esta parte ficou dedicada para as CSS). Exemplo: Usar a tag <address> faz com que o texto fique em itálico mas a principal função da tag é sinalizar que o elemento armazenado é um endereço que pode receber interação de aplicativos de mapeamento por exemplo.

Não utilize tags antigas que estão parando de receber suporte. Mesmo que elas ainda funcionem, podem parar de uma hora para outra.

# {Atalho} | Envelopar trecho de código dentro de uma tag

Selecione o trecho do código a ser envelopado. Use o atalho Ctrl + Shift + P. Digite “abb” para buscar o termo correto e selecione “Wrap with abbreviation” (Envelopar com abreviação). Por fim escreva a tag a envelopar e aperte enter

[!] Fica muito mais rápido envelopar o código assim em poucos passos pois caso o comando seja frequentemente usado, ele sempre aparece como primeiro dentre as opções. Exemplo: Seleciona o trecho do código > Ctrl + Shift + P > Enter (a opção envelopar é a primeira a aparecer) > Digita a tag que terá o trecho como conteúdo > Enter

# {Atalho} | Aumentar ou reduzir espaçamento da indentação no código

Selecione todo o trecho do código a ser modificado

* Ao utilizar a tecla **Tab**, o trecho de código ganha mais espaçamento mantendo a indentação.
* Se quiser reduzir o espaço do código mantendo a indentação, use o atalho **Shift + Tab**.

# {Atalho} | Escrever vários trechos de código ao mesmo tempo

**Clique + Tecla Alt** nas posições que deseja editar.

# {Atalho} | Criar várias linhas com as mesmas tags

Utilize “**\*[QuantidadeDeCópias]**” para criar várias réplicas da sua tag.

Ex:

**<p>\*4** Cria 4 linhas de parágrafos.

**<h2>\*3** Cria 3 linhas de Título Lvl 2

# {Atalho} | Mover código para cima ou para baixo sem recortar e colar

Selecione o trecho de código desejado > Segure a tecla ALT > Use as setinhas do teclado para cima ou para baixo e o trecho do código será deslocado sem sobrescrever os seus vizinhos.

# {Atalho} | Pedir ao VSCode para criar a página HTML declarada que ainda não existe

Exemplo: Ao escrever um trecho de código para gerar um hyperlink que leva o usuário até a página 2 do site e o arquivo dessa página ainda não foi criada, você pode colocar o cursor do mouse em cima, segurar a tecla CTRL + clicar em cima do nome da página (pagina02.html).

O VSCode mostrará um erro ao tentar acessar a página inexistente mas exibirá a opção de cria-la.

# -=-=-= Links e Âncoras no HTML5 =-=-=-

**Âncora/Hyperlink** -> Área clicável em um site que serve para direcionar a navegação do usuário para outra página.

# -=-=-= Link externo =-=-=-

**Link externo** -> Link que aponta/direciona para outra página web. Esta estratégia é indicada para manter a aba do nosso site ativa após o usuário querer acessar um link externo. Caso fosse um link interno, o indicado torna-se apenas sobrescrever a página antiga para evitar transtorno de abas desnecessárias.

**Como usar**: Utilize a Tag **<a>** de Anchor (Âncora) envelopando o trecho do código que se tornará um hyperlink. No parâmetro **href”**[Link]**”** escreva o link para onde o usuário será direcionado ao clicar.

[!] Deste modo a Tag Âncora sobrescreve o site do hyperlink com a nova página visitada. Para fazer com que a página do hyperlink abra em uma nova aba, utilize os parâmetro s **target=”\_blank”** e **rel=”external”** dentro da tag de abertura da âncora.

# -=-=-= Link interno =-=-=-

Direciona para outra página do nosso site. Ou seja, para criar outra página, faça outro arquivo .html armazenado na mesma pasta que a página principal.

# Criar um hyperlink que direcione para outra página web sua

Use a tag **<a href’[NomeDoArquivoHTMLdaOutraPag]’>** para transformar um trecho de texto em um hyperlink que abra o arquivo da outra página.

**[\*]** Da para indicar para os mecanismos de busca qual é a próxima página e qual é a anterior através dos parâmetros **href=”next”** e **href=”prev”**respectivamente**.**

**[\*\*]** Exemplo de patrocinador sem deixar vínculo ou indexação com o site que contém o hyperlink: <p> Esta página é um oferecimento de <a **href=”**youtube.com**” target=”\_blank” rel=”**nofolow**”**>Youtube</a> </p>. O parâmetro NoFollow indica ao mecanismo de busca para não indexar/vincular o site ao outro.

**[\*\*\*]** Caso vá acessar uma página do seu próprio site, podemos colocar o **target=”\_self”** para indicar ao mecanismo de busca que a página a ser acessada também é do próprio site.

# -=-=-= Link para acessar página em pasta anterior =-=-=-

O parâmetro <a>href=”**./**index.html”</a> indica que a página está indicando que a página está na própria pasta.

Para retornar para uma página em uma pasta anterior, use o parâmetro <a>href=”**../**index.html”</a>. Deste modo até o atalho **Ctrl + Espaço** reconhece e indica a página que você talvez esteja procurando em uma pasta anterior.

# -=-=-= Criando links para download =-=-=-

Ao salvar os arquivos que serão baixados dentro de alguma pasta do seu site, utilize a tag **<a href=””>** contendo o endereço do arquivo para download.

[\*] Caso ainda não tenha o arquivo ou um endereço para preencher este parâmetro, use uma **#** para indicar que o link está vazio!

# Configurar link para fazer download do arquivo ao clicar

Adicione na tag **<a href=”[EndereçoDoArquivo]”>** os parâmetros **download=”[NomeDoArquivo]”** e **type=”[TipoDoArquivo]”**.

**[!!]** Para saber o tipo correto do arquivo, procure na lista do seguinte site: <https://www.iana.org/assignments/media-types/media-types.xhtml>.

# -=-=-= Imagens dinâmicas (mostra imagens de tamanhos diferentes de acordo com o tamanho da tela) =-=-=-

**Passo 1 | Criando imagem no tamanho desejado**

Utilizando o Gimp, vá em arquivo > Nova para criar uma imagem.

Medidas utilizadas nos exemplos do Professor Guanabara:

\*Todas com 80 de resolução, perfil de cores RGB e precisão inteiro de 8 bits.

- Para celular: 300x300px;

- Para tablet: 700x700x;

- Para notebooks: 1000x1000x;

**Passo 2 | Escrevendo para identificar tamanho das imagens**

Clique na ferramenta de texto, mude a cor para preto clicando na caixinha de cor e clique para começar a escrever.

Para colocar o texto no centro da página, clique em uma das réguas e arraste a linha. No canto inferior esquerdo aparece a posição dela para te guiar. Coloque o texto na posição que seja igual a metade do tamanho da pag.

Mude a opacidade do texto para 50%.

**[\*]** Para aplicar o mesmo texto alinhado nas outras páginas: clique na camada do texto, arraste para a outra aba e solte na página. Ela ficará alinhada no centro com as mesmas propriedades, pronta para ser renomeada.

**Passo 3 | Mude a cor do fundo para diferenciar as imagens do fundo do navegador**

Selecione a camada fundo > escolha a ferramenta do baldinho no canto superior esquerdo > Mude para a cor que preferir > clique no fundo da página para aplicar

**Passo 4 | Utilize a tag <picture> anotada na sessão “Tags HTML5”**

# -=-=-= Introdução às CSS =-=-=-

# Não utilizar parâmetros de css (ainda mais obsoletos) junto com o código HTML

Além de poluir o código, ainda dará muito mais trabalho de editar e atualizar o estilo dos grupos de elementos da interface.

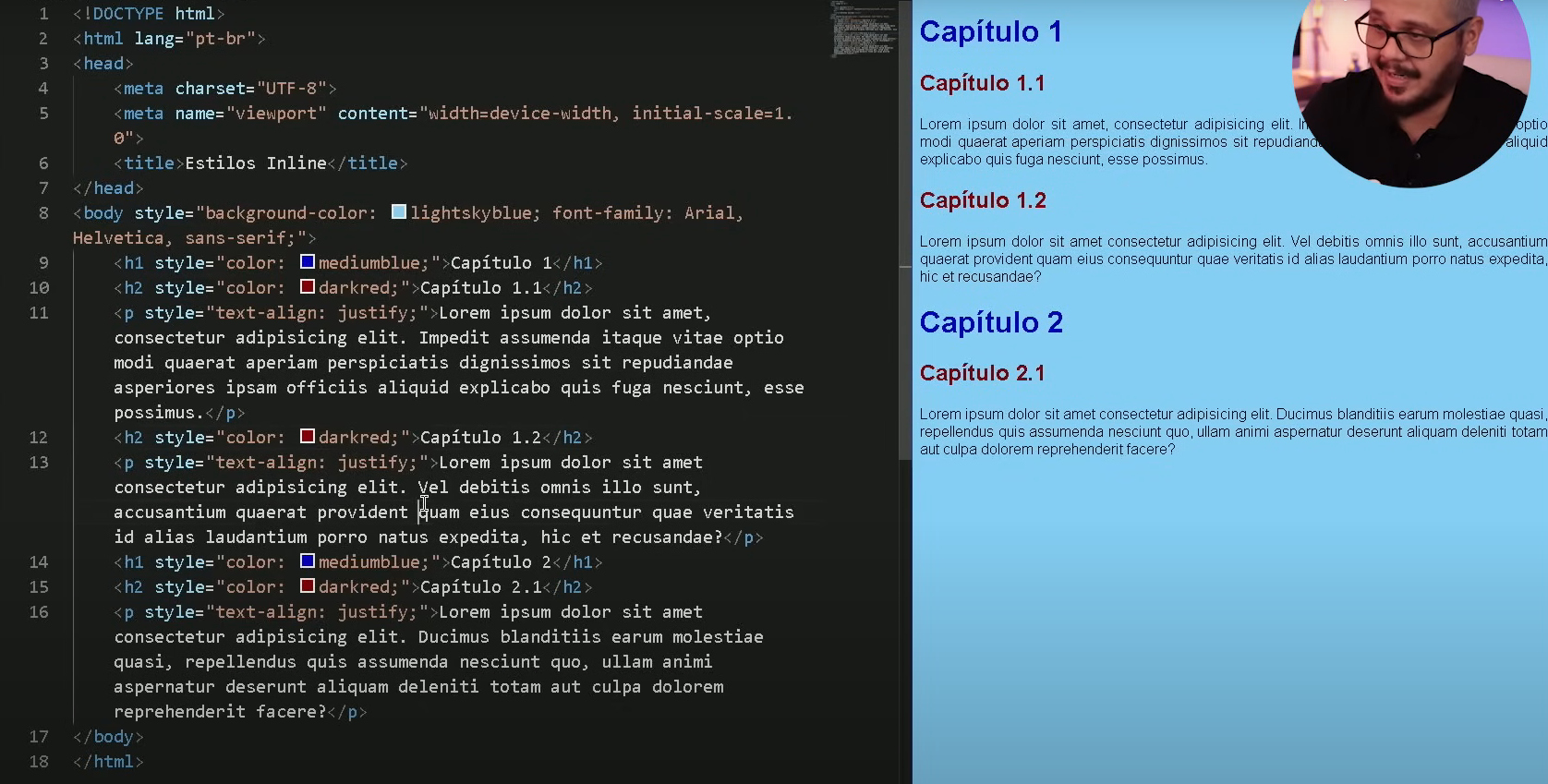
Estes trechos de código utilizando elementos obsoletos que estejam estilizando a página web fora da conformidade serão indicados pela marcação vermelha na cor da fonte.

[\*] É possível utilizar diferentes tipos de estilos (InLine, Interno e Externo) no mesmo código.

# Estilos InLine (estilo na mesma linha da tag)

Modo trabalhoso de se estilizar uma página web configurando argumentos de estilo dentro das estruturas em HTML. O mais recomendado é descrever os parâmetros de estilo dentro do arquivo contendo os códigos CSS com os seus devidos comandos.

Para aplicar um estilo na mesma linha de código, escreva os parâmetros dentro da tag desejada. Exemplos de tags em que pode ser aplicado: body, h1, p, etc.



**Desvangagens:**

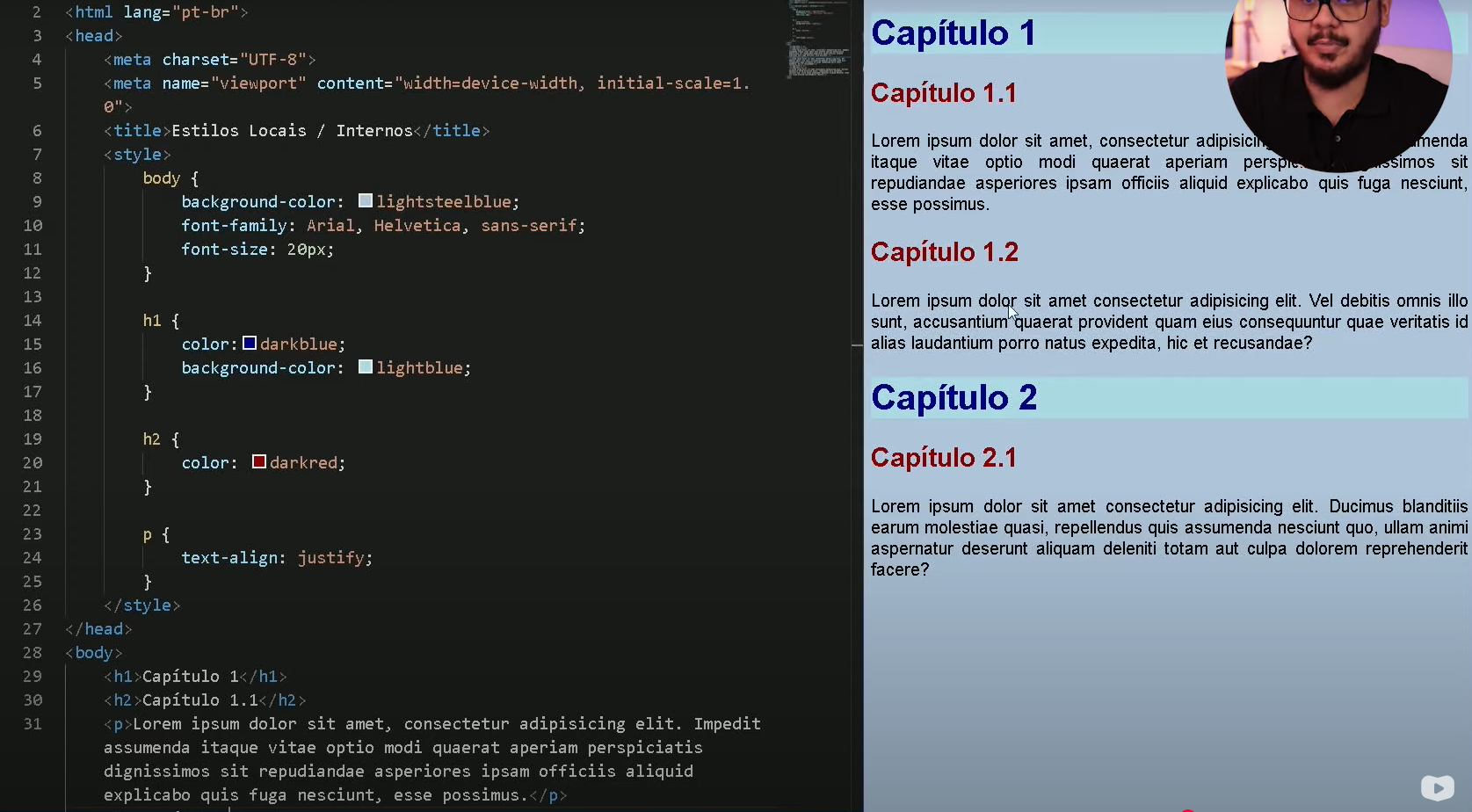
- Código poluído com conteúdo e estilo misturados.

- Alteração do estilo de literalmente cada tag do código manualmente.

# Estilos Locais / Internos

O código CSS ficará armazenado na sessão head dentro do próprio arquivo contendo o código em HTML.

Para editar os estilos criamos a tag “style” e usamos seletores para indicar quais elementos queremos personalizar.



Os parâmetros de estilo virão dentro das chaves que podem ser separadas ao quebrar a linha.

**Desvantagens**:

- O trecho de estilo CSS ocupa muito espaço inicial do código.

- Para aplicar o estilo de uma página nas outras, precisaríamos copiar e colar o trecho em CSS em cada uma delas. Para editar também.

# Estilos Externos

Em vez de precisar editar ou copiar/colar o código de estilo para várias páginas manualmente, existe uma maneira bem mais prática de manipular as personalizações em um arquivo próprio para ela, sem deixar nenhum código CSS nos arquivos de conteúdo do site.

**Como usar**: Crie um novo arquivo chamado **[nome].css** (por padrão o nome é style.css).

Para acessar este arquivo, escreva em cada página a tag **“style:css”**> o código ficará escrito como **<link rel=”stylesheet” href=”[nome].css”>**.

Na nova página CSS, você pode adicionar uma regra ao começo do código para configurá-la. Exemplo: @charset= “UTF8;” para manter a compatibilidade de acentuações e caracteres da língua portuguesa.

Por fim, cole os seletores e parâmetros em CSS dentro do arquivo.

[?] Podem existir mais de uma folha de estilos sendo utilizadas ao mesmo tempo para alternar quando quiser ao mudar o indicador da folha a ser utilizada dentro da página html. Neste caso os parâmetros delas se somarão.

# -=-=-= Lista de parâmetros para estilos em CSS =-=-=-

[\*] Os parâmetros abaixo precisam estar contidos dentro do trecho **style=””** para funcionarem.

# background-color (cor de fundo)

Define a cor de fundo da página.

# Color (mudar cor)

Altera a cor do elemento/objeto onde o estilo foi inserido.

# font-family (trocar a fonte)

Muda a fonte utilizada em textos para o estilo de fonte para a indicada.

# text-align (alinhamento)

Ajusta o alinhamento do texto de acordo com a opção selecionada.

Opções disponíveis:

* Left | Esquerda
* Right | Direita
* Center | Centralizer
* Justify | Justificar
* Start | Começo
* End | Fim

# text-indent (espaçamento inicial)

Aumenta o espaço de início do parágrafo. Unidade medida em px (pixels).

# text-decoration (sublinhar, negrito, etc)

Adiciona efeitos como sublinhar, negrito, linha superior, riscado no centro, etc.

# background-image (degradê/gradiente)

Requer uma função interna para indicar as cores e o sentido do gradiente.

- **linear-gradient**: forneça primeiro a direção, por exemplo:”to [right, left, top, bottom]” ou “90deg” para escrever a medida em graus. Separe por vírgula e escreva quantas cores desejar usar, começando pela cor de origem até a final. Exemplo:

[\*] background-image: linear-gradient(to top, white 30%, blue)

\*Como o tamanho padrão do gradiente é pequeno, se ele estiver na vertical (to top ou bottom) precisa-se mudar seu tamanho para ajustar na página. Como por exemplo usando uma configuração global (\*) height.

**[?]** É possível fornecer também um valor de porcentagem após cada cor para aumentar o diminuir seu espaço de ocupação na tela.

- **radial-gradient**: Faz um gradiente radial. Em vez de fornecer a angulação do gradiente, será fornecido o formato.

[\*] Exemplo: radial-gradient(circle, cor1, cor2, cor...)

# \*{} Configuração global

Altera as propriedades contidas entre chaves para todos os estilos. Deve ser escrita dentro da tag style!

Exemplo: <style>

\* {

height: 100%}

</style>

# background-attachment

Pode receber o valor “fixed” para manter o plano de fundo do site fixo

# main

Funcionamento semelhante às tags <div> para repartir o conteúdo. Esta, porém, indica conter o conteúdo principal.

# paddin [top, right, bottom, left] (espaçamento até a borda)

Também chamado de acolchoamento. Sua unidade de utilização é em pixels (px).

[!] Dica do Guanabara seguir a ordem das direções do padding como se fosse o ponteiro de um relógio: cima > direita > abaixo > esquerda.

**[\*]** Atalho shorthand para implementar: padding [padding-top] [right] [bottom] [left]. Caso todas as medidas sejam iguais, pode declarar apenas a primeira variável e ela será aplicada para as demais. Se colocar apenas duas medidas, ele aplica a primeira para cima e baixo e a segunda para esquerda e direita.

# margin (tamanho de margem)

Determina o espaçamento dos elementos em HTML da margem. Pode receber o valor “auto” para automaticamente espaçar simetricamente as margens da tela.

[\*] Utilize o comando “margin: auto;” para que os espaçamentos da margem sejam definidos automaticamente.

**[\*]** Atalho shorthand para implementar: margin [margin-top] [right] [bottom] [left]. Mesmo funcionamento do padding.

# outline (linha/contorno externo da margem)

Espaçamento presente logo após o limite da margem do conteúdo.

**[\*]** Atalho shorthand para implementar: outline [outline-width] [outline-style] [outline-color]

# box-shadow ou text-shadow (sombra atrás de um objeto/elemento do HTML5)

Primeiro parâmetro define o deslocamento da sombra para o lado; Segundo parâmetro define o deslocamento da sombra para baixo; Terceiro parâmetro ajusta o quanto a sombra vai espalhar; O último parâmetro configura a transparência da sombra.

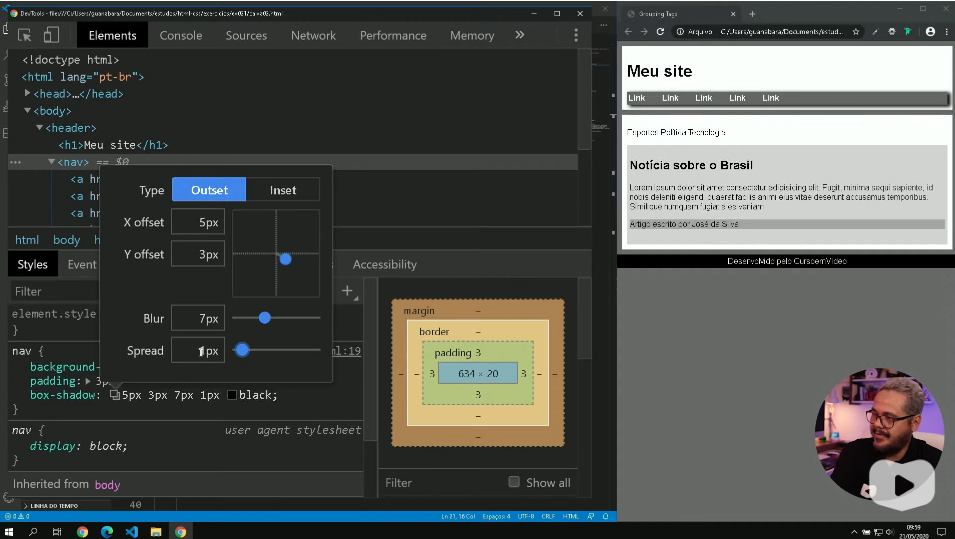
Se a forma em HTML estiver com os cantos arredondados, a sombra acompanhará o formato.

**[?] Formula short-hand**: “box-shadow [DeslocamentoHorizontal] [DeslocamentoVertical] [Espalhamento] [CorDaSombra]”

Também é possível adicionar sombra interna (inset) ou externa (outset ou deixe em branco) adicionando um destes parâmetros logo após o trecho “box-shadow”.

Exemplo: box-shadow 10px 10px 15px [TomMaisEscuroQueDaPaleta]. Este tom escurecido que o tom escuro da paleta fará uma sombra mais discreta por ter uma cor semelhante ao do fundo. Outra ideia interessante é adicionar uma transparência na sombra para ela ficar mais discreta ainda (adicionando valor Alfa como RGBA).

**[\*]Dica para adicionar sombras usando o Inspecionar do navegador**: Ao inspecionar alguma página de navegador e selecionar no código o elemento que deseja adicionar sombra, coloque o cursor sobre os ícone dos 3 pontinhos e selecione a opção “Adicionar sombra/Add shadow”. Use a interface para configurar as usas preferência, copie e cole o código no seu editor para implementá-lo.



# span (apenas um separador de código)

Identifica o trecho contido dentro da tag para receber algum atributo ou classe.

# div (separador de uma mesma página)

Cria divisões no conteúdo de uma página de ponta a ponta horizontalmente.

# border-width (Largura da borda)

Altera a largura da borda de um elemento em HTML.

# border-height (Altura da borda)

Altera a altura da borda de um elemento em HTML.

# border-style(Estilo da borda)

Recebe os valores solid (preenchimento inteiro/solido), dotted (pontilhado), dashed (serrilhado), double (linha dupla), groove (interior e exterior da borda em tons diferentes causando leve efeito 3D), entre outros.

# border (borda de limite da margem)

Ao traçar uma linha ao redor do conteúdo é chamada de parte de conteúdo ou borda(border).

**[\*]** Atalho shorthand para implementar: border [border-width] [border-style] [border-color]

Exemplo: border 10px solid darkgray

# border-radius (arredondamento dos cantos)

Aumenta ou diminui o arredondamento dos cantos de elementos do HTML5. As medidas utilizadas podem ser em pixels ou em porcentagem.

**[\*]** Atalho shorthand para implementar: border-radius [SuperiorEsquerdo] [SuperiorDireito] [InferiorEsquerdo] [InferiorDireito]

Caso utilize apenas duas medidas, ele aplicará nos cantos superior esquerdo e inferior direito.

# border-image-source (borda personalizada/em imagem)

Para aplicar uma borda personalizada, primeiramente crie uma borda simples (utilizando a tag border). Depois, utilize o comando abaixo:

Na declaração “border-image-source:” adicione logo em seguida o parâmetro “url(‘NomeDaImagem.Formato’)” para definí-la.

Use os comandos border-image-slice e border-image-repeat para repartir a borda em seguimentos onde a imagem será aplicada > Ajuste a medida da imagem usando a ferramenta “inspecionar” na página web > Modifique as quantidades de repartição (slice) e largura da borda para ajustar ao seu agrado.

**[\*]** Atalho shorthand para implementar as 3 configurações da borda em apenas uma linha: border-img: url(‘Nome.FormatoDaImagem’) [QuantidadeDoSlice] [Repeat ou Stretch]. O atalho ainda precisa de uma borda criada para aplicar a imagem, mesmo que seja uma borda com cor transparente.

# border-image-slice (recortar borda enorme em seguimentos)

Use border-image-slice: [Quantidade] para definir em quantas partes a imagem usada como borda será repartida para não ficar apenas como uma única imagem esticada.

# border-image-repeat (repetir imagem da borda em cada seguimento)

Utilize border-image-repeat: [Comportamento] para repetir a imagem da borda personalizada em cada seguimento.

Comportamentos que podem ser utilizados:

- Repeat: Repete a imagem em cada segmento da borda.

- Stretch: Estica a imagem para compor a borda inteira.

# -=-=-= Tags de agrupamento =-=-=-

Tags para separar e organizar o conteúdo do site de maneira semântica

# header (Cabeçalho)

Títulos e informações introdutórias.

# footer (Rodapé)

Finalização de conteúdo, informações adicionais e de contato.

# nav (Navegação)

Contendo botões de navegação e links.

# main (Principal)

Corpo principal do conteúdo do site.

# article (Artigo)

Desenvolvimento do conteúdo sobre determinado tópico.

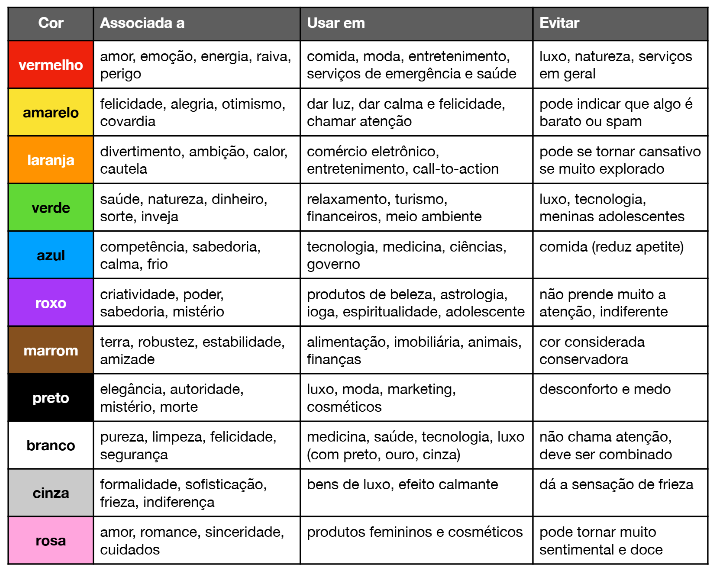
# aside (aparte)

Detalhamento sobre o conteúdo do artigo.

# -=-=-= Psicologia das cores =-=-=-

Escolher um bom de conjunto de cores que possuem harmonia entre si é muito relevante para cativar a atenção do usuário e transmitir certas sensações.

Abaixo há uma tabela com indicações de contextos para utilizar certas cores mas não é uma regra definitiva. Existem vários exemplos por aí de marcas que utilizam combinações de cores fora dos padrões e ainda assim nos impressionam.



# Representação de cores (nomes, hexadecimais, xxxx, xxxxx)

Nomes: style”color= blue”;

Hexadecimais: style”color=#0000ff”;

Possui 16 valores indo do 0 ao 9 + do A ao F. As duas primeiras casas informam quantidades de vermelho, as duas do meio dizem a quantidade de verde e as duas do final ditam a quantidade de azul. 0 representa o mínimo (nada) daquela cor enquanto o F representa o máximo de quantidade daquela cor (valor 255).

RGB: style”color= rgb (0, 0, 255)”;

Informa a distribuição das cores Red (Vermelho), Green (Verde) e Blue (Azul) para compor as demais cores. Valor mínimo é 0 e o máximo 255.

HSL (Hue[Matiz], Saturation [Saturação] e Lightness [Luminosidade]);

A matiz contabiliza indica a cor a ser utilizada. No caso do azul é 240.

A Saturação e a Luminosidade ditam o brilho ou escurecimento da cor em porcentagens de 0% a 100%.

**[?]** O VSCode fornece caixas de manipulação das cores ao clicar e arrastar para mudar a cor, intensidade, tonalidade e transparência. É possível trocar também a unidade de medida clicando no título que contém a receita da cor exibida.

Caso adicione algum valor de transparência, o RGB vira RGBA e o HSL vira HSLA pela presença deste valor Alpha.

# -=-=-= Harmonia de Cores: Circulo cromático e combinações =-=-=-

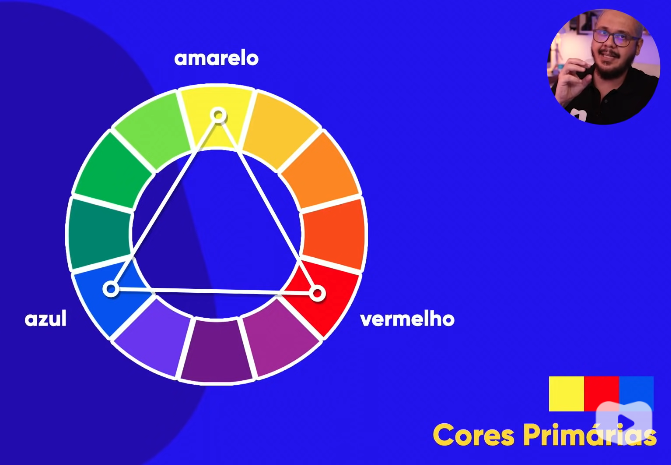
# Circulo Cromático

Distribuição de cores que indica certas combinações complementares, de contraste, harmônicas, etc.

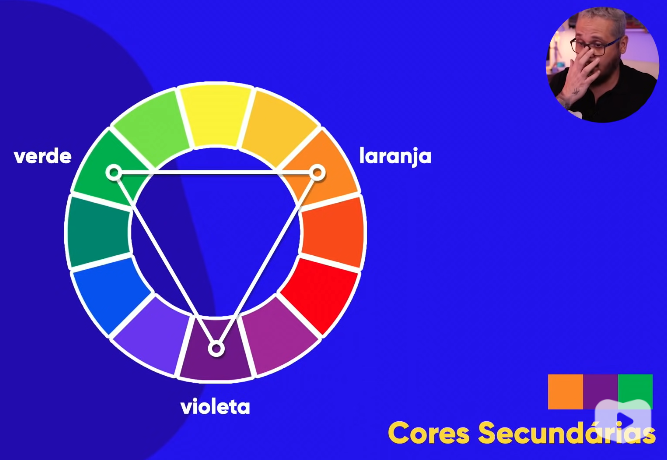
O ideal é buscar a simetria das cores: Ou seja, determinadas combinações proporcionam melhor sinergia, complemento, contraste, entre outros elementos entre as cores usadas.

# Cores primárias (Amarelo, Azul e Vermelho )

Combinação de cores sem suavidade entre si.



# Cores secundárias (Laranja, Violeta e Verde)



# Cores terciárias

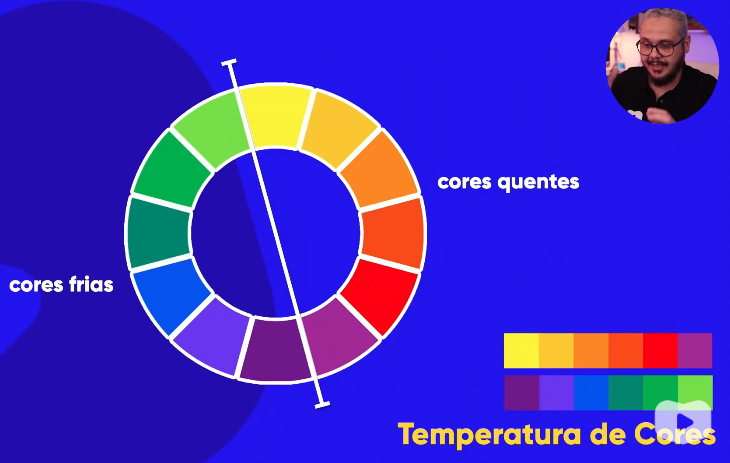
Mistura das cores primárias com as cores secundárias. Seu nome é composto primeiro pela cor primária utilizada seguido pela cor secundária que o distingue. Exemplos: Amarelo-esverdeado, Amarelo-Alaranjado, Vermelho-Alaranjado, Vermelho-Arroxeado, Azul-Arroxeado, Azul-Esverdeado.

Podem ser transformadas em tons pastéis.



# Temperatura de cores

Divididas em cores frias e quentes.



# Paleta de cores

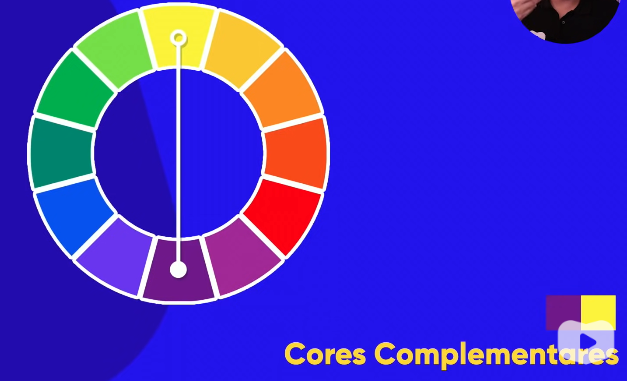
A paleta de cores de um site é muito importante para impactar positivamente nas sensações transmitidas aos usuários. De acordo com o objetivo ou ramo de negócio do cliente, certas cores são mais recomendadas de se usar, mantendo em mente o contraste e quais outros tons combinam melhor com elas.

Recomendado: de 3 a 5 cores. Geralmente 4. Desconsiderando o branco e preto que já fazem parte de todo site.

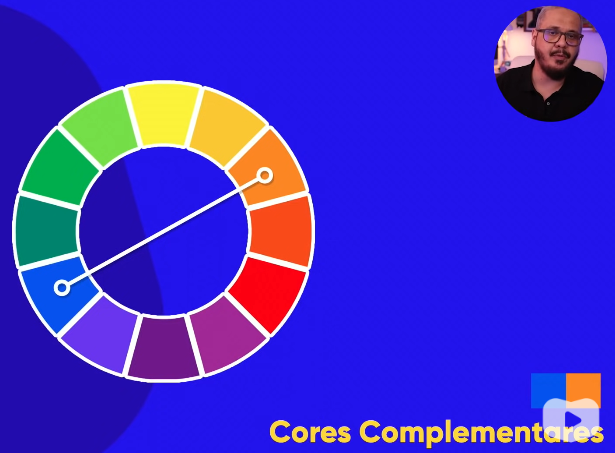
Observar qual a cor primária da logo do cliente para manter a identidade do negócio.

# Cores complementares/opostas (contraste)

Apresentam melhor contraste entre si. Não necessariamente significa que farão combinações bonitas.

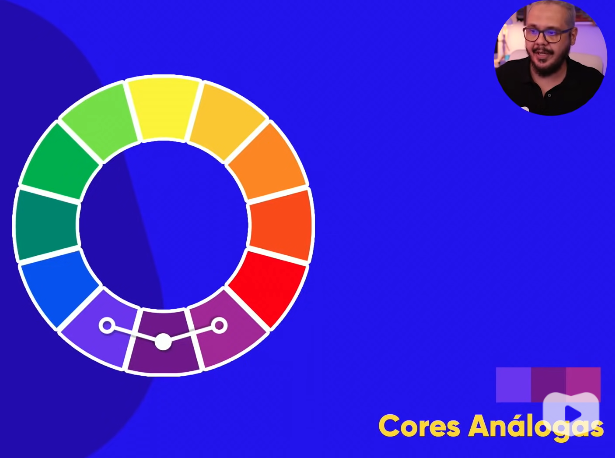






# Cores análogas

Cores vizinhas que causam um pouco de efeito degradê. Não possuem muito contraste entre elas mas ainda possuem boa percepção.



# Cores análogas + cor complementar (resultado bonito)

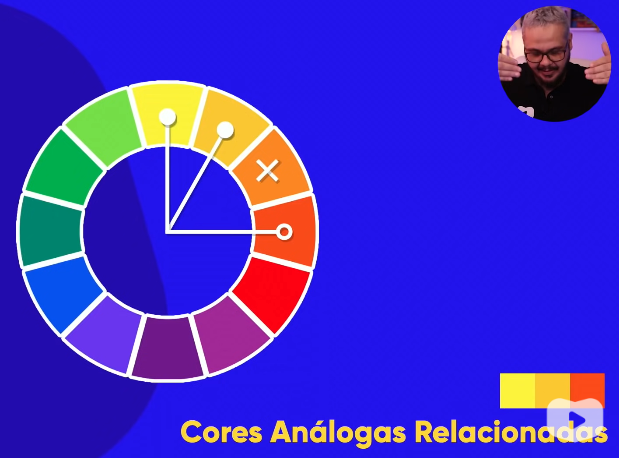
Une cores harmônicas entre si oferecendo também uma opção de cor com ótimo contraste à cor principal.





# Cores análogas relacionadas (Harmonia + Pequeno contraste bonito)

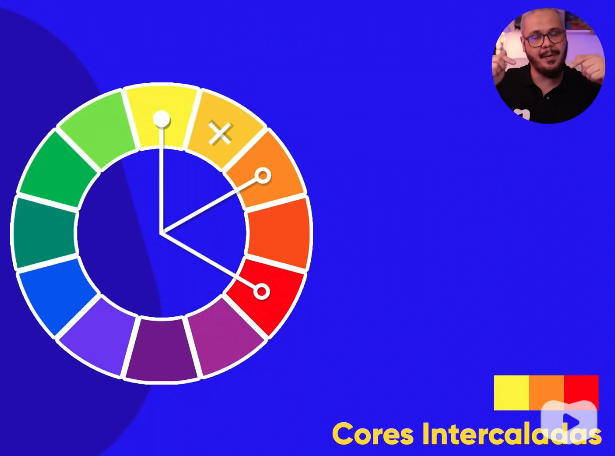
Escolha duas cores vizinhas para manter um leve efeito de degradê com harmonia, pule a próxima casa ao lado e selecione a cor da quarta casa.

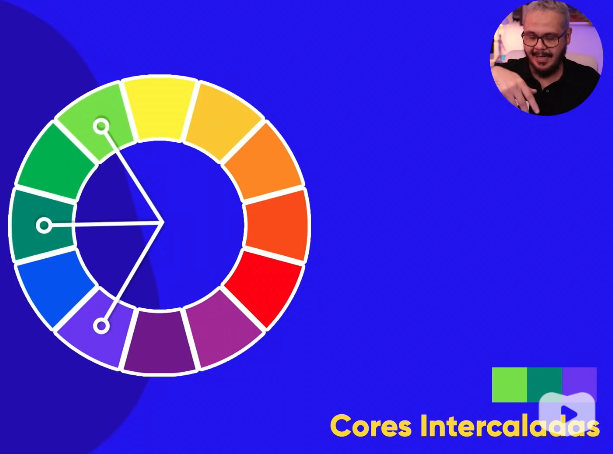




# Cores intercaladas

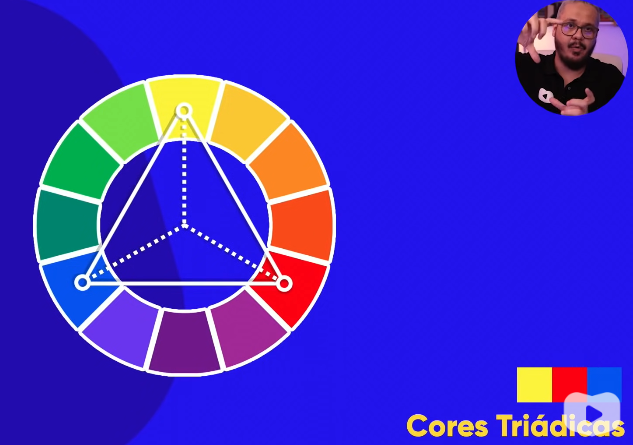
Escolha cores sempre pulando uma casa entre elas. Formam paletas de cores condizentes mas sem suavidade entre seus tons.





# Cores triádicas

Escolha a sua cor primária, pule 3 casas e escola as outras duas cores. No fim a paleta formará um triângulo equilátero. Obs: Cores primárias e secundárias são também cores triádicas.



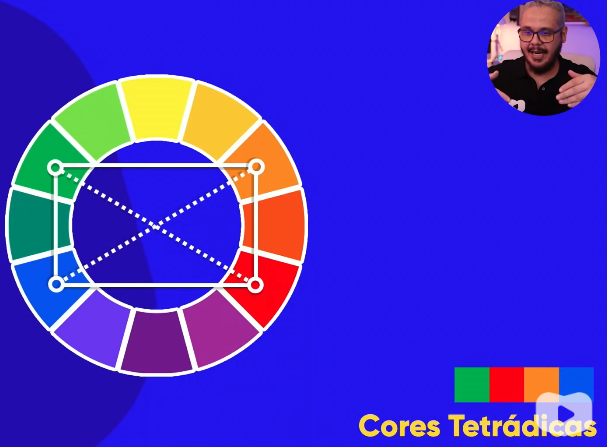
# Cores em quadrado

Pule apena duas casas entre as cores escolhidas. Ao escolher 4 cores, o seu alinhamento formará um quadrado. Fornece cores variadas entre primárias, secundárias e terciárias. Gera uma combinação de cores fortes e balanceadas.



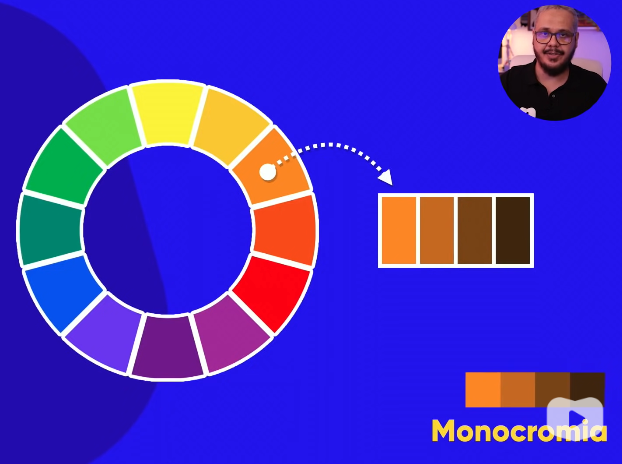
# Cores tetrádicas

Escolha uma cor qualquer, pegue a sua cor complementar (diretamente oposta). Pegue outra cor principal e novamente escolha a sua cor complementar.



# Monocromia (degradê)

Trabalha com apenas uma cor alterando a sua saturação e o brilho/luminosidade.



# -=-=-= Ferramentas para trabalhar com cores =-=-=-

**Color.adobe.com**

Ferramenta gratuita para elaborar paleta de cores. Grande parte dele é gratuito, só não dará para salvar sem a assinatura adobe.

Manipular paleta: Escolha a unidade de medida para a paleta > Selecione uma das opções de combinações de cores (monocromática, complementar, triádica, etc)

\*Pode inserir o código de alguma cor fornecida para automaticamente mostrar as cores que combinam com ela. Permite também copiar o código em diversas medidas das cores selecionadas.

Extrair tema: Analisa uma imagem fornecida (como uma logo, por exemplo) e fornece a paleta de cores dela com ajustadores. Mostra os códigos do resultado na aba “Disco de cores”

Descobrir paletas: Selecione a aba “Explorar” > Filtre por “Exibir” > Temas de cores > Mais usado.

**Paletton.com**

Além de interagir com o circulo cromático e as suas combinações, proporciona também algumas outras funcionalidades interessantes.

Mostrar algumas exibições da paleta: Aba Preview > Color Scheme

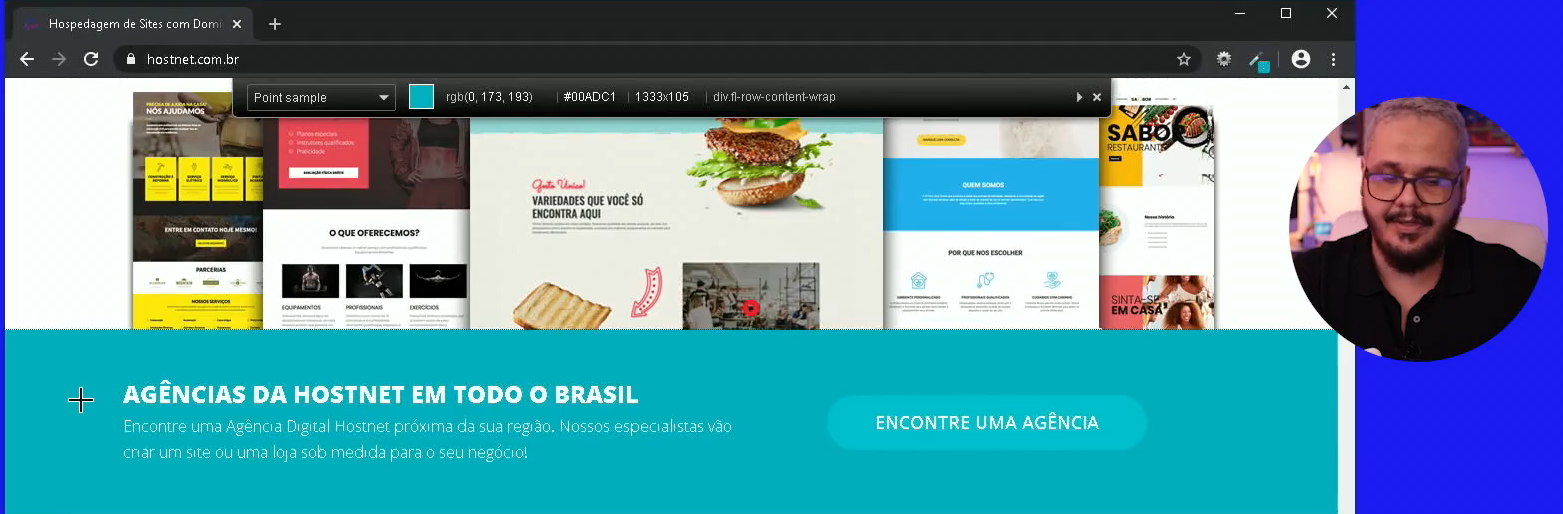
Visualizar simulação de cores no site: Aba “Examples” - Monta um site de modelo para conferir como fica a organização das cores nele. Possui exemplos de páginas com tema claro ou escuro para visualizar cabeçalhos, rodapés, blocos de conteúdo, entre outros elementos.

**Coolors**

A cada vez que apertar a barra de espaço o algoritmo sugere uma nova paleta de cor interessante. Pode selecionar as cores que gostar e travar para gerar outras combinações com elas.

**Indicação de extensão do google para capturar código de cor em página web**

Acesse a loja de extensões do google > Procure pela extensão “ColorZilla” > Opção “Page Color Picker” > Clique na cor para o cabeçalho te mostrar os códigos dela



# -=-=-= Tipografia =-=-=-

Tipo > Týpos > Escrita

Grafia > Impressão

É o estudo da maneira como gravamos e escrevemos símbolos legíveis que possibilitem uma fácil compreensão e transmitam sentimentos pertinentes ao contexto que estão inseridos.

# -=-=-= Anatomia dos tipos =-=-=-

Indicações:

- Para textos de corpo muito longo, utilizar fontes sem serifa. Precisa utilizar cores complementares (com alto contraste).

Estuda os espaçamentos horizontais e verticais; o sentimento transmitido de uma determinada fonte.

Exemplo: Times New Roman | Fonte serifada da sub-classe romana.

- Glifo: Cada caractere ou letra de uma fonte ou família de fontes.



- Fonte: Todos os glifos de uma determinada família com estilo específico.



- Família tipográfica: Variações da fonte de uma mesma família.



- Categoria de fonte: Conjuntos de fontes com determinados elementos presentes ou ausentes que alteram o sentimento transmitido.

-- Legenda: Sans-serif (sem serifa); Monoespaçada (cada caractere tem a mesma largura); Handwriting simula a caligrafia humana como se tivesse sido feita à mão; Fonte display é apenas decorativa sem seguir muitas regras de tipografia.



- Altura x (curiosidade): Toda fonte nova é feita baseando-se na “altura x”, ou no caso, o tamanho da letra x minúscula, no qual todas as outras letras minúsculas seguirão como guia de suas alturas.

- Altura das maiúsculas: Altura máxima das letras maiúsculas.

- Ascendente: Espaçamento da altura cujo tamanho vazou para cima das linhas da altura x.

- Descendente: Espaçamento da altura cujo tamanho vazou para baixo das linhas da altura x.

- Corpo: Medida da altura total de uma fonte, desde o extremo inferior até o superior considerando letras maiúsculas ou minúsculas.



[!] Nem todas as fontes possuem todos os elementos descritos.

**Anatômicos geométricos**

- Serifa: Pequenos tracinhos ou prolongamentos de letras da fonte que auxiliam o nosso cérebro a percorrer os alinhamentos durante a leitura.

- Haste: Segmento vertical que da estrutura para a letra.

- Filete: Segmento horizontal que une duas hastes ou arcos.

- Arco: Parte curvada da letra para unir hastes.

- Esporão: Segurar ou criar uma base da letra.

- Vértice: Pontinha isolada na extremidade da letra.

- Terminal: Pedaço da letra que não está ligado a mais nenhum outro elemento.



**Anatômicos corporais**

- Braço: Elemento que se projeta de uma haste e se projeta para cima.

- Perna: Elemento que se projeta de uma haste ou braço e se projeta para baixo.

- Pé: Base que segura a fonte. Geralmente onde existem serifas.

- Espinha: Curvatura específica da letra S que possui duas.

- Barriga: Curvatura que se projeta e se conecta na mesma haste.

- Olho: Espaço interno de uma letra que é rodeado de todos os lados por elementos de uma letra.

- Orelha: Sobra ou pontinha presente em algum canto superior de uma letra.

- Cauda: Rabinho que se projeta na parte inferior de uma letra.

# -=-=-= Familias de fonte com CSS =-=-=-

**[!]** Novamente reforçando que fontes sem serifa proporcionam melhor leitura em textos, ainda mais se estes forem grandes.

Declaração safe-combination: Ao declarar qual fonte será utilizada no estilo da página, caso aplique a fonte Arial no VSCode, ele colocará também as fontes Helvetica e Sans-Serif logo em seguida. Deste modo o código testará a primeira fonte e caso ela falhe, o programa testará as demais fontes como redundância e aplicará uma fonte genérica caso todas as demais falhem.

**[?]** Existem no Google algumas ideias de combinações de fontes mais seguras que podem ser utilizadas. Pesquise por “css web safe font combinations” > Site da W3Schools.

Na declaração <style>

body { font-family: [sans-serif/serif/monospace]

}

</style>

Sans-Serif indica ao computador para exibir qualquer fonte padrão sem serifa.

Serif faz a mesma função, porém, com fontes sefiradas.

Monospace também é semelhante mas com fontes monoespaçadas.

**[!!]** Uma boa prática é sempre colocar uma alternativa final como “Sans-Serif” ou “Serif” para indicar o uso de qualquer fonte com ou sem serifa caso todas as outras opções indicadas estejam indisponíveis.

# -=-=-= Tamanho de fonte e suas medidas =-=-=-

**Medidas absolutas**

Cm, mm, in (inches/polegadas), px (pixel), pt (ponto), pc (paica)

**Medidas relativas**

Em (tamanho maiúsculo atual da fonte), ex (altura x de uma fonte), rem (Parecido com o EM mas é relativo ao tamanho da fonte configurado na raiz/body da página), vw (view width/largura da viewport ou tamanho de tela, em porcentagem), vh (view height/ aktyra da viewport ou tamanho de tela, igual ao view width), % (porcentagem pura).

[\*] Para tamanhos de fontes, é recomendado utilizar apenas os pixels (px) e os em.

# -=-=-= Peso, estilo e shorthand font =-=-=-

# Peso da fonte (grossura)

O **peso** da fonte refere-se à grossura de seus glifos (letras). Algumas famílias tipográficas possuem mais gradientes de peso enquanto fontes mais comuns possuem apenas versões normais e negrito.

A variável que escolhe a fonte com certo peso é “font-weight” com parâmetros como (lighter/normal/bold/bolder). Também pode ser passado um valor entre 100 e 900 para a variável, porém, não é compatível com todas as fontes.

# Estilo da fonte (itálico, sublinhado, etc)

Outra característica que algumas fontes podem ou não possuir é a de **estilo**. Para isto, a fonte precisa possuir variantes com font-style:”italic” e/ou negrito em diferentes variedades de peso também.

Para aplicar o sublinhado, a fonte precisa ser compatível com a variável “font-decoration: underline”.

# Shorthand (atalho para modificar estilo, peso, tamanho e família em uma única variável)

Shorthand: Propriedade que permite declarar múltiplas definições em uma mesma linha.

**->** “**font: [Font-Style] [Font-Weight] [Font-Size] [Font-Family]**”

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

A variável precisa ser utilizado na seguinte ordem: font-style -> font-weight -> font-size -> font-family. Não é obrigatório que todos os itens da variável sejam declarados. Na falta de algum deles, um valor padrão será utilizado.

# -=-=-= Fontes do Google (Google Fonts) =-=-=-

Endereço: fonts.google.com

# Como importar fonte sem precisar baixa-la

Escolha a fonte desejada no site do Google Fonts > Opção “Get Font” > Get “Embed Code” > Escolha a opção “@import” > Cole o código em HTML no seu código.

[!] O código para importar a fonte precisa vir antes das demais declarações que aplicarão o seu estilo, de preferência na primeira linha após a tag de abertura <style>. Pode também importar vários estilos que a fonte possua.

[!!] Como a fonte será importada de um link da web, o site precisa estar conectado na internet para funcionar.

# -=-=-= Fontes do Google (Google Fonts) =-=-=-

Como utilizar fontes externas/baixadas: Use o seletor especial @font-face. Ao aplica-lo no VSCode, ele virá com a estrutura montada para indicar a font-family e a url respectivas do arquivo da fonte.

Estrutura:

@font-face{

font-family: [NomeDaFamiliaDeFonteQueVocêEstáCriando];

src url ([NomeArquivoFonte1.{Extensão}]) format ()

}

Se quiser adicionar outra fonte ou arquivo de fonte, replique o trecho “src url ([NomeArquivoFonte2.{Extensão}]) format ()”

A variável format pode receber os parâmetros opentype(otf); truetype(ttf); embedded-opentype; truetype-aat (AppleAdvanced Typography); svg.

Por fim, ao criar esta nova família de fontes utilizando uma fonte personalizada, use alguns parâmetros para especificar as suas propriedades, como: font-weight, font-style, etc.

**[\*]** Indicação de site para encontrar fontes: DaFont. Possui catálogos e exibições das fontes, seus glifos e características.

**[!]** Algumas fontes podem não possuir caracteres acentuados ou suporte para estilos em negrito/itálico.

Ao baixar ou receber o arquivo de uma fonte, extraia a pasta compactada. Dentro podem haver arquivos .otf e/ou .ttf. Copie as fontes e cole numa pasta específica para armazenar as fontes do seu site.

**[!]** Dependendo do navegador, alguns podem lidar melhor com arquivos de fonte .otf e outros com fontes .ttf. Uma boa ideia é a de utilizar ambos os formatos.

# -=-=-= Capturando fontes usadas em sites da web =-=-=-

🡪 Para detectar fontes textuais

- Indicação de extensão do Google: Fonts Ninja.

Mostra detalhadamente a composição de cada fonte textual (exceto as que fazem parte de imagens) como: nome da fonte utilizada, seu tamanho, espaçamento, efeito, cor e espessura. Permite também manipular algumas propriedades para testar.

🡪 Para detectar fontes dentro de imagens

Indicações de sites:

- WhatFontIs: Melhor inteligência de detecção. Permite enviar a imagem para detecção, aumentar o contraste para ajudar a distinguir as letras e separar os caracteres para serem analisados individualmente.

\*Nem toda fonte ou site terá detecções de fonte que funcionem 100% ou todas as vezes.

FontSquirrel(Aba Font Identifier): Arrastando a imagem o site pede para selecionar qual fonte será identificada e para separar os caracteres que ajudarão a identificar.

MyFonts: Mesmo passo a passo dos anteriores de subir a imagem, selecionar trecho da fonte a ser analisado e separar os caracteres.

# -=-=-= Usando identificadores (IDs) nos elementos da HTML =-=-=-

**[!]** Servem apenas para um elemento único. Utilizar IDs iguais em elementos distintos funciona mas é a maneira errada de ser fazer!

Para usar adicione o parâmetro **id=”[NomeIdentificador]”** dentro da tag de abertura a ser identificada.

Para estilizar uma tag que possua ID, escreva o identificador dela seguido por # e o nome do identificador. Também funciona se escrever apenas a hashtag e o nome do identificador.

Exemplo:

**h1#[NomeIdentificador]{**

color: gold;

**}**

**[?]**

Tudo o que for id em HTML, é referenciado com # nas CSS.

Tudo que for classe em HTML, é referenciado com . nas CSS.

# -=-=-= Diferenças entre ID e Class =-=-=-

[\*] Sempre dê nomes dos identificadores de acordo com a sua função (ex: principal, intermediário, etc) e não pela sua forma (grande, verde, rapido, etc) pois a sua forma pode ser alterada e o nome estará incorreto.

IDs servem para identificar elementos únicos. Utilizar a mesma ID para elementos distintos é a maneira errada de referenciar elementos! O ID sobrepõe as definições declaradas por uma classe.

Classes também servem para identificação e podem ser utilizadas para vários elementos. Pode ser utilizada mais de uma classe nos elementos HTML separando simplesmente com um espaço sem vírgulas.

**[?]** Uma boa prática é escrever apenas o ponto e a classe para ser utilizada onde mais desejar.

Exemplo: Em vez de h2.principal, escreva apenas .principal.

# -=-=-= PseudoClasses (:) =-=-=-

São relacionadas ao estado de um elemento e servem para atribuir comportamentos de acordo com o estado dos elementos. Para isto, precisam de uma classe ou elemento html para serem utilizadas.

Tipos de estados (ex: ativo, marcado, vazio, habilitado, etc)

- :Hover | atribui um certo comportamento ou característica ao colocar o cursor em cima do elemento html atribuído a esta pseudoclasse.

- :Visited | atribui um comportamento a ser exibido quando o link referenciado já foi acessado/visitado.

- :Active | Exibe um comportamento quando o elemento é clicado ou acionado.

# -=-=-= PseudoElementos (::) =-=-=-

Manipula os conteúdos dos elementos HTML. Diferente das pseudoclasses que manipulam a estética.

- ::After | Pode ser usado para adicionar conteúdo (parâmetro content) ou outras funcionalidades após o conteúdo do elemento original.

- ::Before | Mesmo funcionamento do After, porém, o Before adiciona elementos antes do original.

# -=-=-= Elementos filhos de outros elementos (>) =-=-=-

Use o símbolo > para indicar que o elemento a direita é filho do elemento à esquerda.

Exemplo: **div > p** | O parágrafo é filho de uma div.

Pode ser utilizado como classe ou pseudo classe também. Exemplo -> div:hover > p

# -=-=-= Conceitos do modelo de caixas =-=-=-

# Caixa (box)

Todo elemento visível de um site em uma tela é uma caixa(box). Cada caixa possui uma altura height e uma largura width.



**[Dica]** Clique com o botão direito em uma página web e vá na opção “Inspecionar...” > Na aba elementos serão exibidas várias medidas e propriedades os elementos em HTML.

# Aninhamento

Conceito de estruturas contidas uma dentro da outra formando uma hierarquia.

# Borda (border)

Ao traçar uma linha ao redor do conteúdo é chamada de parte de conteúdo ou borda(border).

# Padding (Acolchoamento ou preenchimento)

Espaço interno adicional entre o conteúdo contido dentro da box e o limite da borda. Calculado em qualquer dimensão (cima, baixo, esquerda, direita).

# Margin (Margem)

Espaçamento externo da borda até um determinado limite.

# Outline (Contorno ou Traçado)

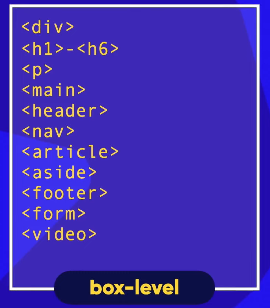
Espaçamento por fora da borda que a contorna.

# -=-=-= Tipos de caixas =-=-=-

# box-level

Novos conteúdos sempre são criados em novas linhas abaixo. Seu conteúdo sempre reserva toda a largura disponível da página na tela.

Exemplo de tags que utilizam box-level:



# inline-level

Caixas de novos conteúdos sempre são criados utilizando apenas a largura necessária para encaixar este conteúdo. Começam a partir do mesmo parágrafo de outros conteúdos anteriores sem quebrar linha.

Exemplo de tags que utilizam inline-level:



# -=-=-= Modelo de caixas na prática =-=-=-

**[Dica]** Clique com o botão direito em uma página web e vá na opção “Inspecionar...” > Na aba elementos serão exibidas várias medidas e propriedades os elementos em HTML.

\*É possível modificar os valores na inspeção da página web. As alterações ocorrerão apenas na exibição do usuário sem alterar o código original.

User agent: Atuação do navegador exibindo o conteúdo por conta própria. Está presente, por exemplo, ao abrir uma página web que não possui definições de CSS descritas mas ainda assim aparece com código HTML configurado pelo navegador.

A variável display vem por padrão block, referente a exibição de caixa no nível box-level. Ela pode receber outros valores como Flex que serão estudados mais adiante. Existe também o display inline que exibe o conteúdo no espaço da mesma linha sem quebrá-la caso haja espaço.

# -=-=-= Ferramenta para planejar layouts de sites =-=-=-

Mockflow – Como usar:

Crie um novo projeto > Selecione o tipo de layout a ser desenvolvido, neste caso deste estudo de HTML será Site

Coloque uma estrutura “Browser Frame” para simular uma página web e configure as suas dimensões.

Crie um retângulo com a mesma largura do Browser Frame > Configure a altura de sua preferência. Este retângulo será o espaçamento do cabeçalho.

Pode adicionar um “Menu Bar”(ou outro retângulo) logo abaixo para o menu contendo opções de navegação.

Crie retângulos ou quadrados para simular o espaçamento do trecho principal de conteúdo (main) e o rodapé do site.

Para simular conteúdos da para adicionar “títulos”, caixas de textos com escritas Lorem Ipsum automaticas, imagem, video-players, etc.

- Mock: Parecidos com wireframes, são esboços da estrutura do layout de um site sem se preocupar com características personalizaveis (ex: cor).

# -=-=-= Utilizando variáveis globais em HTML5 =-=-=-

Use a pseudoclasse :root (raiz) que todo documento possui. É nela que todas as variáveis globais seram declaradas para utilizar em qualquer outro documento ou página do projeto.

Para declarar variáveis globais dentro da root use ”--” antes do nome da variável. Exemplo:

:root{

--[NomeDaVariável]: [Parâmetro];

--cor0: #c5ebd6;

}

# Variáveis globais para cores

Caso utilize variáveis globais para salvar cores definidas por você, ao declarar a cor que será utilizada em algum elemento css, aperte CTRL + Espaço para ver a lista de cores. Abaixo das cores padrões estarão listadas as cores que você definiu.

**[\*]** Outra vantagem das variáveis globais utilizadas para definir cor é que ao alterar a cor da variável global, todos os elementos que a utilizam serão automaticamente atualizados também.

# Variáveis globais para fontes

Exemplo:

:root {

--fonte-padrao: Arial, Verdana, Helvetica, sans-serif;

}

# -= Usando seletor global (\*) para modificar todos os elementos HTML =-

Sua principal utilização é para definir parâmetros padronizados para todos os elementos HTML sem precisar selecioná-los um por um.

Exemplo:

\* {

margin = 0px;

}

Todos os elementos de HTML5 terão suas margens zeradas por padrão.

# -=-=-= Adicionando responsividade ao projeto do site =-=-=-

Defina uma largura máxima e mínima para o site. Na largura máxima, o restante da página será preenchida apenas com espaçamentos de fundo enquanto o conteúdo continua com um tamanho bem distribuído. Na largura mínima, o conteúdo é adaptado para fornecer uma visualização melhor.

Utilize o “Inspecionar” na página web. Ao aumentar ou diminuir o tamanho da janela do navegador, será informado no canto da tela o tamanho atual dela.

O objetivo é evitar o vazamento de conteúdo e a rolagem lateral quando a tela está pequena demais e parágrafos muito compridos quando a tela está grande demais.

# {Plug-In} | Simular o tamanho da página em telas de celular, tv, etc

Window Resizer: Informa o tamanho atual da página, modifica o tamanho da página de acordo com o padrão de algum outro dispositivo (tv, celular, tablet, etc).

# -=-=-= Aprendendo a utilizar Git e GitHub =-=-=-

Repositório: Conjunto de arquivos armazenados na máquina local (repositório local) ou em um servidor de maneira remota (repositório remoto).

Git: Aplicativo responsável por salvar localmente as versões do código desenvolvido.

GitHub**:** Aplicativo dedicado para armazenar remotamente o seu repositório em um servidor/na nuvem.

Push: Refere-se ao ato de subir para salvar arquivos atualizados no repositório local para o repositório em um servidor.

Ao utilizar um computador pessoal ou de qualquer outro lugar, tendo o git e github instalados, você pode fazer login na sua conta e clonar (fazer uma cópia) dos arquivos do seu repositório ou de outras pessoas. Use a interface do GitHub Desktop para abrir o código pelo seu editor de código preferido e abra também o local onde os arquivos estão armazenados na máquina.

Após modificar os arquivos da maneira que preferir, o GitHub Desktop apontará as modificações que foram realizadas. Deste modo você pode fazer um Commit para publicar as alterações contendo título e descrição que foram realizadas. Desta maneira o seu repositório local será atualizado. Lembre-se também de subir/upar (Push) as alterações no repositório remoto do GitHub para que as outras pessoas tenham acesso.

Quando finalizar, pode deslogar da sua conta no GitHub Desktop, apagar o nome de usuário e endereço na Aba “Options” > “Git”. Antes disso, pode remover e enviar o repositório local para a lixeira também pois ao fazer um Push (Upar/subir) os arquivos para o repositório remoto, já haverá uma versão salva na núvem.

# -=-=-= Usando imagens de fundo na página =-=-=-

Utilize a função background-image: **url (EndereçoDaImagemDeFundo)** para aplicar uma imagem que será repetida até preencher o elemento html em que foi inserida caso a imagem seja pequena. Se for adicionada uma imagem grande, ela será aplicada com partes da imagem faltando a depender da proporção dela.

\*O endereço da imagem pode ser tanto a localização do arquivo da imagem quanto realmente uma url da web.

Estas imagens inseridas não possuem significado ou sentido semântico. São apenas elementos visuais de fundo.

Vale ressaltar que se forem usadas imagens muito pesadas, o site ficará lento para carregá-las e o consumo de rede aumentará.

# -=-=-= Propriedades para configurar imagem de fundo =-=-=-

As seguintes propriedades são utilizadas para configurar a tag background-image.

[\*] Procure sempre definir uma cor de fundo para o seu site, mesmo que o fundo da página seja uma imagem pois ao acessar um site, primeiro é carregado o conteúdo textual e depois de alguns segundos chegam as mídias. Portanto, se o usuário já quiser imediatamente consumir o conteúdo, ele poderá fazê-lo sem ver uma página branca genérica.

url (EndereçoDaImagem): Ao utilizar o endereço do arquivo da imagem armazenado no computador ou na web, o parâmetro url faz com que a imagem seja usada como fundo do elemento ao qual está inserida. A imagem também será replicada para preencher todo o espaço disponível para ela. O tamanho será fixo, então caso o tamanho da tela mude, as imagens permanecem com o mesmo tamanho.

Background-size (largura e altura em pixels): Determina o tamanho da imagem ao ser aplicada como fundo.

- *Auto*: Por padrão o parâmetro vem com o valor “auto” que define o seu tamanho de acordo com o espaçamento disponível.

- *Contain*: faz com que a imagem seja 100% exibida de fundo, sem cortá-la.

- *Cover*: Cobre a imagem toda sem deixar nenhum espaçamento vazado. Em troca, pode podar algumas extremidades da imagem dependendo da sua proporção e o espaço disponível. Porém, caso o conteúdo da página seja extenso, pode ser que a imagem acabará e aparecerá o fundo da página web.

Background-repeat: Altera o comportamento da repetição da imagem através dos seguintes parâmetros:

- Repeat: Faz a imagem se repetir. Este já é o comportamento padrão do background-image.

- no-repeat: Adiciona apena uma imagem. O ideal é a imagem estar com o tamanho dimensionado para encaixar na página. O ponto inicial para o posicionamento da imagem é por padrão no canto superior esquerdo.

- repeat-x ou repeat-y: Repete a imagem apenas no eixo X ou Y de acordo com qual deles foi declarado.

Background-position [left|right|center] [top|bottom|center]: Define o posicionamento da imagem de fundo dentro do elemento HTML onde está inserida. Por padrão o posicionamento já vem como Left Top ao aplicar uma imagem de fundo utilizando a tag background-image.

Background-attatchment [Scroll | Fixed]: Define se o fundo a página rolará junto com ela ou permanecerá parado enquanto apenas o conteúdo da página é rolado.

**[\*]** Atalho shorthand para implementar a configurações de fundo:

Background **[color] [image] [position] [repeat] [size] [attatchment]**

Exemplo: background **black url(‘imagem.png’) center center no-repeat cover fixed**;

# -=-=-= Alinhamentos vertical e horizontal =-=-=-

- **Horizontal**

Utilize “margin: auto;” para centralizar horizontalmente um elemento HTML em relação à largura da tela.

- **Vertical (e Horizontal também)**

Para alinhar verticalmente um elemento em HTML, precisa-se de outro elemento que o conterá (no exemplo ele está com id=’container’). O elemento contido pode ser identificado com o id=’conteudo’.

No elemento ‘container’, ele receberá o parâmetro “‘position: relative’” enquanto o elemento ‘conteudo’ receberá “’position: absolute’”.

Ao declarar o elemento conteúdo com posicionamento absoluto, torna-se possível determinar outros dois parâmetros: top e left para definir o espaçamento à esquerda e ao topo do elemento conteúdo em relação ao seu container.

Como este posicionamento será feito no ponto superior esquerdo do objeto conteúdo, ele ainda não estará no centro. Apenas a sua ponta. Para ajustá-lo use o parâmetro:

# Transform: translate(-50%, -50%);

Esta definição posicionará o objeto metade para cima e metade para esquerda, consequentemente centralizando o objeto conteúdo no centro do objeto container onde está contido.