Sumário

[Anotações HTML 3](#_Toc109244393)

[Definição de URL 3](#_Toc109244394)

[Index 4](#_Toc109244395)

[Código Base HTML 4](#_Toc109244396)

[Símbolos Especiais 4](#_Toc109244397)

[Emojis 5](#_Toc109244398)

[Tipos de Imagens 5](#_Toc109244399)

[Ícone de Favorito 5](#_Toc109244400)

[Imagens de Domínio Público 5](#_Toc109244401)

[Parágrafo de teste 6](#_Toc109244402)

[Tags Obsoletas, como saber quais são 6](#_Toc109244403)

[Encapsular 6](#_Toc109244404)

[Vários Cursores no VScode 6](#_Toc109244405)

[Listas 6](#_Toc109244406)

[Links Internos e Externos 7](#_Toc109244407)

[Links Para Downloads 7](#_Toc109244408)

[Imagens Dinâmicas 8](#_Toc109244409)

[Áudios 8](#_Toc109244410)

[Vídeos 9](#_Toc109244411)

[Grouping Tags e Semantics Tags 9](#_Toc109244412)

[Anotações CSS. 11](#_Toc109244413)

[Criar um arquivo CSS 11](#_Toc109244414)

[Psicologia das Cores 11](#_Toc109244415)

[Representação das Cores em CSS 12](#_Toc109244416)

[Harmonia das Cores 12](#_Toc109244417)

[Paleta de Cores 13](#_Toc109244418)

[Criação de Paleta 13](#_Toc109244419)

[Criação de Degradê 13](#_Toc109244420)

[Escolha de Fonte 14](#_Toc109244421)

[Tamanho da Fonte 15](#_Toc109244422)

[Peso, Estilo e ShortHand da Fonte 16](#_Toc109244423)

[Utilizar Fontes do Google 17](#_Toc109244424)

[Utilizar Arquivos de Fontes 18](#_Toc109244425)

[Identificar Fontes de Outros Sites e Imagens 18](#_Toc109244426)

[Alinhamento e Formatação de Texto 19](#_Toc109244427)

[Formatando Listas 19](#_Toc109244428)

[Seletores Personalizados Id e Class 20](#_Toc109244429)

[Pseudo-classes e Pseudo-elementos 21](#_Toc109244430)

[Modelos de Caixas: Block-Level e Inline-Level 22](#_Toc109244431)

[Configurações de Caixa em CSS e ShortHand 23](#_Toc109244432)

[Sombras em Caixas 25](#_Toc109244433)

[Bordas Arredondadas nas Caixas 26](#_Toc109244434)

[Montagem da Estrutura do Site 26](#_Toc109244435)

[Criando Variáveis em CSS 26](#_Toc109244436)

[Site Responsivo 28](#_Toc109244437)

[Planos de Fundo 29](#_Toc109244438)

[Centralização de Conteúdo na Vertical 30](#_Toc109244439)

[Tags HTML 33](#_Toc109244440)

[Entendendo o CSS 36](#_Toc109244441)

[Programas e Extensões do Chrome 37](#_Toc109244442)

[Programas 37](#_Toc109244443)

[Extensões do Chrome 37](#_Toc109244444)

**HTML Anotações, Comandos e Tags**

# Anotações HTML

## Definição de URL

Todo site tem um IP que é usado para acessa-lo e que muda constantemente. E é para isso que existe o **DNS** **(Domain Name System)**, ou sistema de nomes de domínio. Eles são como grandes “listas telefônicas”, criando uma ligação entre o nome do site que é a **URL** **(Uniform Resource Locator)** e o número de IP relacionado a ele.

A parte principal dessa URL é o **domínio**. O domínioé um nome único que vai conseguir identificar o seu servidor ou as suas páginas.

As terminações, como .com, .gov.br, .io, .edu.br. indicam tecnicamente que o site é de uma instituição comercial, governamental, educacional, ONGs, artistas, etc. Além disso, alguns desses ainda indicam o país (.br). Esses são os **TLD (Top Level Domain).**

* **GTLD**: São TLDs genéricos, sem indicação de país. Como .com, .net, .gov, .org, .io, .info, .online, .store, etc.
* **ccTLD**: São TLDs com designação do país (*coutry code*). Como .com.br, .edu.us, .co.fr, .jp, .es, etc.

O que vem antes do domínio é chamado de **subdomínio.** Por exemplo o site do google tem vários subdomínios:

* **www**.google.com.br - Dá acesso ao site principal do Google
* **images**.google.com.br - Você vai para o Google Imagens
* **maps**.google.com.br - Acessa o Google Maps
* **mail**.google.com.br - Entra no Gmail

Além do subdomínio e domínio, uma URL também é composta pelo protocolo utilizado, que pode ser http:// ou https://, dependendo se o seu servidor tem ou não segurança por **SSL** (um serviço de criptografia de dados).

Uma URL também pode ter um **caminho**, que indica pastas a percorrer para achar um arquivo específico. Por exemplo: [https://www.github.com**/gustavoguanabara/html-css/tree/master/aulas-pdf**](https://www.github.com/gustavoguanabara/html-css/tree/master/aulas-pdf)

****

## Index

Quando for criar o arquivo inicial HTML uma dica é sempre colocar o nome do arquivo como **“index.html”** tudo em minúsculo “*qualquer arquivo deve ser nomeado com letras minúsculas”*. Index significa índice e é usado como arquivo da pagina inicial de um site.

## Código Base HTML

Para criar o **código base do arquivo HTML** no VScode basta colocar exclamação “!” e apertar enter.

## Símbolos Especiais

**Símbolos especiais:**

|  |  |  |
| --- | --- | --- |
| **Símbolo** | **Descrição** | **Entitiy** |
|  | *Espaço em branco* | &nbsp; |
| ® | *Marca registrada* | &reg; |
| © | *Copyright* | &copy; |
| ™ | *Trade Mark* | &trade; |
| € | *Euro* | &euro; |
| £ | *Libra (pound)* | &pound; |
| ¥ | *Yen* | &yen; |
| ¢ | *Cent* | &cent; |
| ∅ | *Vazio* | &empty; |
| ∑ | *Soma* | &sum; |
| Δ | *Delta* | &Delta; |
| ← | *Seta esquerda* | &larr; |
| ↑ | *Seta acima* | &uarr; |
| → | *Seta direita* | &rarr; |
| ↓ | *Seta baixo* | &darr; |

Há muitos outros símbolos em: <https://www.w3schools.com/charsets/ref_utf_symbols.asp>

Quando o símbolo estiver no site com apenas o número de referência, coloque antes **&#**, o número e termine com “**;**”. Exemplo: para fazer o sinal < o comando fica **&#60;** e o sinal > fica **&#62;.** Podemos fazer esses dois sinais também &lt; e &gt;.

## Emojis

Outra coisa que você pode adicionar às suas páginas são os **emojis,** onde cada símbolo possui um código Unicode. Para uma pesquisa mais precisa e atualizada, acesse o site da [www.emojipedia.org](http://www.emojipedia.org) . No site ache o emoji que quiser desça um pouco na página e copie seu codepoints sem o U+, e cole no seu arquivo HTML substituindo o U+ por **&#x.** Exemplo: [😱 U+1F631](https://www.emojipedia.org/emoji/%F0%9F%98%B1/) no HTML fica &#x1F631.

## Tipos de Imagens

Tipos de **imagens** para usar em um site são **JPEG** e **PNG**. Arquivos JPEG são bem compactados e arquivos PNG aceitam transparência. Um programa gratuito para editar imagens é o **Gimp.**

Para colocar uma imagem, o ideal é não ultrapassar 1500 linhas de largura, e resolução de 72. Claro que pode variar, mas quanto maior a imagem, maior será volume de banda e maior o custo de hospedagem.

## Ícone de Favorito

Arquivos **ICO** são usados para colocar como **ícone de favorito**, no site [www.iconarchive.com](http://www.iconarchive.com) há vários para escolher e é possível criar seu próprio arquivo de ícone em [www.favicon.cc](http://www.favicon.cc), [www.favicon.io](http://www.favicon.io).Para colocar o ícone no site vá no arquivo HTML em dentro do campo “*head”* acima da tag “*title”* escreva **link ache favicon** e aperte enter, apague favicon.ico que está entre as aspas do href segure *Ctrl* e aperte *espaço* para achar o caminho do arquivo e colocá-lo entre aspas.

## Imagens de Domínio Público

Existem vários sites que oferecem **imagens de domínio público** para serem usados em seus projetos. Mas a dica que te dei antes continua valendo: leia atentamente as instruções para uso dessas imagens. Algumas delas deixam você usar a arte apenas se o artista for creditado em um texto localizado no seu site. Alguns dos sites que eu mais gosto de visitar quando quero usar imagens com domínio público são: • UnSplash • Pexels • FreePik • Rawpixel • Pixabay • Libreshot • Wikimedia Commons

## Parágrafo de teste

Para criar um **parágrafo de teste** com palavras aleatórias basta escrever **loren** e apertar enter entre as tags <p> </p>.

## Tags Obsoletas, como saber quais são

Algumas **tags antigas do HTML4 se tornaram obsoletas em HTML5**, são tags que não seguem a semântica empregada no HTML5, são as tags de forma que hoje se usa em CSS. Para saber quais a tags estão obsoletas mantenha-se sempre atualizado em <https://dev.w3.org/html5/pf-summary/obsolete.html>.

## Encapsular

Para colocar um **texto entre uma tag de um jeito fáci**l basta selecionar o texto em seguida apertar **Ctrl, shift + P** que abrirá uma barra de busca, nesta barra escreva abb, ache a opção ***“wrap with abbreviation”*** e clique nela. Vai aparecer outra barra de pesquisa, nesta coloque a tag que quiser e aperte enter. *Nota: nas próximas vezes você não precisará mais pesquisar abb, basta apertar Ctrl, shift +p apertar enter e colocar a tag.*

## Vários Cursores no VScode

**Para colocar vários cursores no VScode,** basta segurar o Alt e clicar. Isto é muito útil para editar varias tags ao mesmo tempo.

## Listas

Para mudar o **tipo de uma lista ordenada** basta preencher a tag com o comando *type.* Exemplo: <ol type=”A”>. Tipos: 1, A, a, I, i.

Para mudar o **tipo de uma lista não ordenada** basta preencher a tag com o comando *type.* Exemplo: <ul type=”circle”>. Tipos: disc, circle, square.

E para mudar a **letra ou** **número inicial de uma lista** basta preencher a tag com o comando start. Exemplo: <ol type=”A” start=”5”>. Obs: start é um comando numérico podendo somente ser preenchido com um número, ou seja, se o *type* é *A* e *start* é *5*, a lista se iniciará a partir da letra *E*.

## Links Internos e Externos

Os links podem ser externos ou do próprio site. Para **configurar um link externo**, o ideal é abrir em uma nova guia com o seguinte comando: <a href="................" **target="\_blank"**>. E para **configurar um link interno** do site, o ideal é que abra na mesma guia com o seguinte comando: <a href="................" **target="\_self">.**

**\_blank** abre o link em uma nova guia.

**\_self** abre o link na mesma guia.

Existe um recurso para links usado para indicar qual é a natureza do destino, exemplo: <a href="................." target="\_blank" **rel="external">.**

**next** indica que o link é para a próxima parte do documento atual.

**prev** indica que o link é para a parte anterior do documento atual.

**author** indica que é um link para o site do autor do artigo atual.

**external** indica que é um link para outro site que não faz parte do site.

**nofollow** serve para instruir os buscadores que um determinado hyperlink não deve ser seguido e que sua página de destino não deve receber link juice da página de origem.

## Links Para Downloads

**Para configurar um link de Download** o ideal basta fazer o link diretamente para o arquivo que se deseja efetuar o download e adicionar o atributo *download* com o valor configurado para o nome do arquivo a ser baixado e o atributo *type* para indicar ao navegador que tipo de arquivo está sendo baixado. Exemplo:

<a href=”arquivos/meulivro.pdf” *download*=”meulivro.pdf” *type*=”aplication/pdf”> Baixe aqui o PDF do meu livro </a>

Para saber o que escrever dentro do atributo ***type*** de uma âncora de hypertext, consulte a lista oficial da IANA.org disponível no link abaixo. E para consultar a extensão no site segure Ctrl + F e digite a extensão na barra de pesquisa.

<https://www.iana.org/assignments/media-types/media-types.xhtml>

## Imagens Dinâmicas

Para **criar imagens dinâmicas** basta usar a tag *picture*, com ela é possível criar **imagem ajustável a resolução da tela.** Para isso, é preciso criar várias resoluções de uma mesma imagem e ir adaptando-as aos atributos *source* *meida type* da tag *picture* sempre do menor para o maior.Como exemplo foram usadas 3 imagens. Foto-g com 1000px de largura, foto-m com 700px de largura e foto-p com 300px de largura. Uma dica importante é sempre usar um valor de 50px a mais no atributo de largura.

**<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 flexivel">

**</picture>**

## Áudios

Para **adicionar áudios** ao site usamos a tag *<audio src=””>* podendo usar alguns atributos como: *autoplay* e *controls.* Mas para ter certeza de que vá tocar em qualquer navegador, o ideal é ter o mesmo áudio em vários formatos e configurar a tag de uma maneira diferente, usando o *source src.* Exemplo:

**<audio** preload="metadata" autoplay controls loop**>**

<source src="midias/break-your-lock-and-key.mp3" type="audio/mpeg">

<source src="midias/break-your-lock-and-key.ogg" type="audio/ogg">

<source src="midias/break-your-lock-and-key.wav" type="audio/wav">

<p>Infelizmente seu navegador não consegue reproduzir o áudio. <a href="midias/break-your-lock-and-key.mp3">Clique aqui para baixar o arquivo mp3.</a></p>

**</audio>**

O atributo *preload* indica se o áudio será pré-carregado ou não e aceita três valores:

* *metadata* vai carregar apenas as informações sobre o arquivo (tamanho, tempo, informações de direitos, etc). **Obs: metadata é o ideal para qualquer momento.**
* *none* não vai carregar absolutamente nada até que o usuário clique no botão play ou um script inicie a reprodução.
* *auto* (padrão) vai carregar o arquivo de áudio inteiro assim que a página for carregada, mesmo que o usuário nunca aperte o play.

O atributo *controls* vai apresentar o player na tela. Caso não seja colocado na tag <audio>, o controle será transparente e o usuário não poderá interagir com ele.

O atributo *autoplay*, quando inserido, vai iniciar a reprodução do áudio assim que a página for carregada.

O atributo *loop* vai fazer com que o áudio seja repetido eternamente assim que terminar a sua reprodução.

Uma dica importante é sempre usar arquivos mp3 ou ogg que são mais leves que wav.

## Vídeos

Para **adicionar vídeos** ao site usamos a tag *<video src=””>* podendo usar alguns atributos como: *controls.* Mas para ter certeza de que vá tocar em qualquer navegador, o ideal é ter o mesmo video em vários formatos e configurar a tag de uma maneira diferente, usando o *source src.* Exemplo de vídeo hospedado no próprio servidor: **Obs. Essa opção pode gerar muitos custos devido a largura de banda.**

**<video** controls width="350" poster="imagens/Sem título.png"**>**

<source src="midias/pexels-neve.mp4" type="video/mp4">

<source src="midias/pexels-neve.ogv" type="video/ogg">

<source src="midias/pexels-neve.webm" type="vidoe/webm">

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

**</video>**

Atributos como *controls, autoplay* e *loop* também são compatíveis na tag *video.*

Para **adicionar vídeos sem custos de hospedagem** é possível usar sites próprios para isso, por exemplo, no youtube tem a opção de compartilhar vídeo, nela há uma opção de incorporar que gera o código html para colocar em qualquer site que estiver criando. Veja abaixo um exemplo de código gerado no youtube:

<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" allowfullscreen></iframe>

## Grouping Tags e Semantics Tags

A linguagem HTML padrão tinha apenas duas tags de agrupamento genérico: a **<div>** e a **<span>.** A diferença básica entre elas é que a primeira é um elemento agrupador do tipo **block-level** e o segundo é **inline-level.** No mais, eles agem exatamente da mesma maneira, servindo para juntar vários outros elementos HTML.

Com o surgimento da HTML5, surgiram as tags semânticas de agrupamento. Isso não significa que as <div> e <span> (agora chamadas de não-semânticas) deixaram de existir ou ficaram obsoletas, mas seu uso agora faz menos sentido, pois temos tags para dividir as partes do nosso documento HTML.

Vamos compreender a partir de agora os principais agregadores semânticos da HTML5.

**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.

# Anotações CSS.

## Criar um arquivo CSS

Uma dica para **criar um arquivo CSS** de forma simples é, fazer primeiro o link css abaixo da tag *title*. Vai aparecer uma linha de comando com *href=”style.css”*. Agora segure **Ctrl e clique** em cima de style.css. O VScode vai mostrar um alerta dizendo que o arquivo não existe e se você deseja cria-lo. Clique em criar e pronto.

Todas as **configurações do CSS** são feitas no começo do código. Exemplo: para compatibilidade com o teclado, pode colocar ***@charset******“UTF-8”;.***

|  |  |  |  |
| --- | --- | --- | --- |
| **Cor** | **Associada a** | **Usar em** | **Evitar** |
| **vermelho** | amor, emoção, energia, raiva, perigo | comida, moda, entretenimento, serviços de emergência e saúde | luxo, natureza, serviços em geral |
| **amarelo** | felicidade, alegria, otimismo, covardia | dar luz, dar calma e felicidade, chamar atenção | pode indicar que algo é barato ou spam |
| **laranja** | divertimento, ambição, calor, cautela | comércio eletrônico, entretenimento, call-to-action | pode se tornar cansativo se muito explorado |
| **verde** | saúde, natureza, dinheiro, sorte, inveja | relaxamento, turismo, financeiros, meio ambiente | luxo, tecnologia, meninas adolescentes |
| **azul** | competência, sabedoria, calma, frio | tecnologia, medicina, ciências, governo | comida (reduz apetite) |
| **roxo** | criatividade, poder, sabedoria, mistério | produtos de beleza, astrologia, ioga, espiritualidade, adolescente | não prende muito a atenção, indiferente |
| **marrom** | terra, robustez, estabilidade, amizade | alimentação, imobiliária, animais, finanças | cor considerada conservadora |
| **preto** | elegância, autoridade, mistério, morte | luxo, moda, marketing, cosméticos | desconforto e medo |
| **branco** | pureza, limpeza, felicidade, segurança | medicina, saúde, tecnologia, luxo (com preto, ouro, cinza) | não chama atenção, deve ser combinado |
| **cinza** | formalidade, sofisticação, frieza, indiferença | bens de luxo, efeito calmante | dá a sensação de frieza |
| **rosa** | amor, romance, sinceridade, cuidados | produtos femininos e cosméticos | pode tornar muito sentimental e doce |

## Psicologia das Cores

## Representação das Cores em CSS

As cores podem ser representadas de 4 formas em CSS:

Por **nome**: <h2 style="background-color**: blue**; color: **white**;">Exemplo de cores</h2>

Por código **hexadecima**l: <h2 style="background-color**: #0000ff**; color**: #ffffff**;">Exemplo de cores</h2>

Por código **rgb (red, green, blue)**: <h2 style="background-color: **rgb(0, 0, 255)**; **color: rgb(255,255, 255)**;">Exemplo de cores</h2>

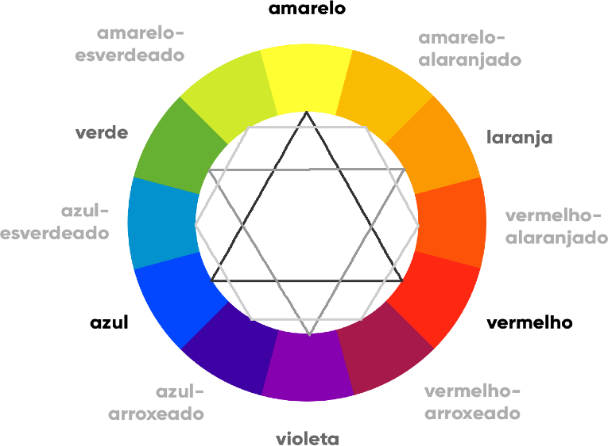
Por características **hsl (hue, Saturation e Lightness):** <h2 style="background-color: **hsl(240, 100%, 50%)**; color: **hsl(0, 0%, 100%)**;">Exemplo de cores</h2>

Todas as representações acima são iguais em cores.

Passando o mouse por cima do nome ou código da cor, abre uma janela de edição da cor, podendo até **colocar transparência.**

## Harmonia das Cores

Imagem do Círculo Cromático



Todos os **detalhes sobre harmonia de cores** estão no Capítulo 13 – Cores em PDF pagina 7 a 9 na pasta do curso. E na aula Harmonia de Cores do modulo 2 do curso HTML com CSS no link: <https://www.youtube.com/watch?v=E2gaDa4ZaTc&list=PLHz_AreHm4dlUpEXkY1AyVLQGcpSgVF8s&index=4>

## Paleta de Cores

**Uma paleta de cores** para criação de um site deve ter em 3 a 5 cores, fora o branco e preto. O mais utilizado é **4 cores**. Se o site for para uma empresa que tem um logotipo o ideal é usar a cor primária deste logotipo também como primária do site.

## Criação de Paleta

Alguns sites para criação de paletas:

<https://color.adobe.com/pt/create/color-wheel>

<https://paletton.com/>

<https://coolors.co/palettes/trending>

No site da adobe além de criar a paleta como quiser ainda é possível extrair cores de um logo clicando em extrair tema. E em explorar tem vários exemplos de paletas prontas para inspiração.

Uma opção para extrair cores de um site é instalando a extensão do chrome **Colorzilla.**

## Criação de Degradê

É possível criar um degradê com a **propriedade** *background-image: ;* junto com o comando interno *“linear-gradient”* (em linha) ou *“radial-gradient”* (circular)*.*

Exemplo:

<style>

body {

background-image: linear-gradient(to right, #c193f5, #2348eb);

}

</style>

Podemos indicar para **que lado o degradê vai** seguir, como no exemplo acima “***to right****”* que significa para esquerda. Exemplos: to right (para direita), to left (para esquerda), to top (para cima), 90deg (90 graus). Se caso usar “*radial-gradient*” usamos o *“circle”* no lugar dos exemplos anteriores para fazer um degradê circular.

É possível colocar quantas cores quiser e selecionar a porcentagem que cada cor ocupa na imagem colocando o valor a frente do código. Exemplo:

background-image: linear-gradient (to right, #c193f5 20%, #2348eb);

Dicas para colocar degradê no “*body”* (cor de fundo)

Usar “*background-attachment: “fixed”* (Deixa o fundo fixo).

Exemplo:

body {

background-image: linear-gradient (to top, #c193f5, #2348eb);

**background-attachment: fixed;**

}

Usar **o seletor global \***

Exemplo:

\*{

**height: 100%;**

}

O comando acima deixa a altura do degrade em 100% se caso colocar de cima para baixo ou de baixa para cima.

## Escolha de Fonte

Para escolher a fonte, usamos a propriedade *font-family: ;* “em seguida apertamos Ctrl e espaço entre : e ;”.

Exemplo:

<style>

body {

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

}

h1, h2 {

font-family: 'Times New Roman', Times, serif;

}

</style>

No exemplo acima note que em body a três fontes, isso porque duas delas são fontes similares e a ultima é a opção sem serifas. Algumas fontes não tem em todos os navegadores então o ideal é fazer um comando com fontes similares. Neste caso o navegador tenta *arial* primeiro, se o navegador não encontrar ele tenta *helvética*, caso não tenha nenhuma das opções ele coloca a fonte padrão do navegador *sem serifas*.

Veja no link abaixo fontes similares para usar:

<https://www.w3schools.com/cssref/css_websafe_fonts.asp>

## Tamanho da Fonte e Todas as Letras em Maiúsculo

Existem vários tipos de medidas possíveis para serem usadas em CSS, que são divididas em medidas absolutas e medidas relativas.

Medidas absolutas:

* cm – centímetro
* mm – milímetro
* in – polegada
* px – pixel
* pt – ponto
* pc – paica (ponto e paica são medidas de impressão em papel não recomendados para conteúdo de tela)

Medidas relativas:

* em – altura da letra M maiúscula da fonte
* ex – altura da letra X minúscula da fonte
* rem – altura da letra M maiúscula que está fixada ao elemento-raiz “r” significa “root”: “root em”)
* vw – porcentagem de largura da viewport, por exemplo, se a largura do navegador é 750px, 1vw equivale a 7.5px.
* vh – porcentagem de altura da viewport, por exemplo, se a altura do navegador é 900px, 1vh equivale a 9px.
* % – porcentagem

De **acordo com o site de padronização do CSS o W3C**, o recomendado para tamanho de fonte é ***px*** e ***em***. Por padrão o **tamanho normal da fonte é 16px, que equivale a 1em.** A medida em *px* pode variar de acordo com o modela da tela também.

Exemplo:

font-size: 1.5em;

No exemplo acima estamos atribuindo *1.5em* a fonte o que equivale a *24px* pois *1em* é = a *16px.*

    text-transform: uppercase; /\*Texto todo em maiúsculo\*/

    font-variant: small-caps; /\*Texto todo em maiúsculo mas as primeiras letras maiores.

## Peso, Estilo e ShortHand da Fonte

Há 4 configurações muito usadas em fontes que são:

* font-style – estilo da fonte. Exemplo: itálico
* font-weight – peso da fonte. Exemplo: negrito
* font-size – tamanho da fonte.
* font-family – família da fonte. Exemplo: Arial

Observação: Algumas fontes não possuem todas as configurações. Como é o caso da **propriedade** *font-weight* que pode ser preenchido como *bold, bolder, lighter*, ou de *100* a *900,* mas a maioria das fontes contidas no VScode não possuem tantas alterações assim.

Podemos usar todas essas configurações separadamente, exemplo:

body {

font-family: 'Work Sans', sans-serif;

font-weight: bolder;

font-size: 3em;

font-style: italic;

}

Mas também podemos usar um **“shorthand”**, que seria todos os valores em uma única propriedade. Para isso basta preencher os valores na ordem correta. A propriedade é ***font: ;*** e a ordem para preenchimento dela é font-style **->** fonte-weight **->** font-size **->** font-family**.**

Exemplo:

h1 {

font: italic bolder 3em 'Work Sans', sans-serif;

}

Nos dois exemplos acima temos o mesmo resultado a diferença é que com o shorthand o comando fica mais curto e mais simples. E lembrando que **não é obrigatório preencher todas as configurações dentro da propriedade** *font*. Por exemplo se quisermos fazer as mesmas configurações do exemplo, mas sem colocar a fonte em itálico basta:

h1 {

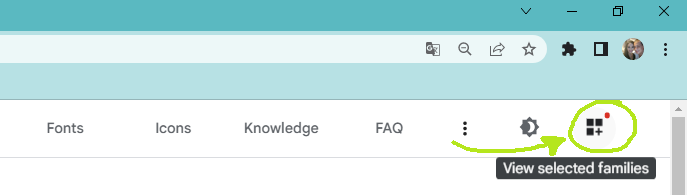
font: bolder 3em 'Work Sans', sans-serif;

}

E para colocar **sublinhado** usamos a declaração *text-decoration: underline;*

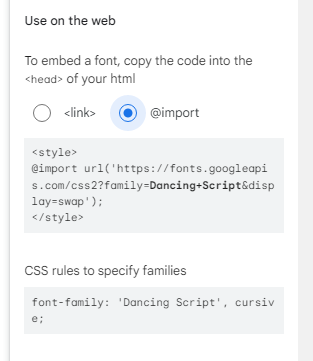
## Utilizar Fontes do Google

No site <https://fonts.google.com/> é possível pesquisar a família de fonte que quiser, copiar o código e colocar dentro do CSS. Com isso, sua página usa a fonte direto do google através de link no CSS, evitando problemas de compatibilidade com o tipo de navegador e também deixa a página mais leve.



Basta fazer uma busca, escolher a fonte e clicar no canto superior direito do site.

Em seguida escolha as variações da fonte que escolheu e clique em +. Depois é só colar o *@import* no começo do documento CSS ou da tag style. O site também dá a declaração para selecionar a fonte.



## Utilizar Arquivos de Fontes

Para adicionar um arquivo de uma fonte específica ou baixado da internet **usamos a regra** ***@font-face.***Nela configuramos o caminho do arquivo, formato do arquivo, configuramos um nome para a fonte e podemos configurar também os padrões *weight* e *style* da fonte. Uma dica é sempre que for colocar o arquivo de uma fonte em um site o ideal é criar uma pasta com o nome “font” ou um nome de sua preferência. Lembrando que o caminho, nome e formato deve ser colocado sempre entre **aspas simples**.

Exemplo:

@font-face {

font-family: 'Filxgirl';

src: url('fonts/Filxgirl.TTF') format('truetype'), url('fonts/Filxgirl.OTF') format('opentype');

font-weight: normal;

font-style: normal;

}

Os **formatos mais comuns** e compatíveis com a maioria dos navegadores são os utilizados no exemplo acima, **TTF e OTF**. Mas há vários tipos de formatos de arquivos de fonte, veja alguns abaixo:

* opentype (OTF)
* truetype (TTF)
* embedded-opentype
* truetype-aat (Apple Advanced Typography)
* SVG

Um site ideal para download de fontes é <https://www.dafont.com/pt/>

## Identificar Fontes de Outros Sites e Imagens

Para **identificar fontes em formato de texto** de um site qualquer, basta instalar a **extensão** do chrome **“*fonts ninja*”.** Com ela é possível ver qual a fonte usada, o código da cor, tamanho e estilo.

Para **identificar fontes que estão em imagens** é possível usar alguns sites:

<https://www.whatfontis.com/> , <https://www.fontsquirrel.com/> , <https://www.myfonts.com/>

O tutorial para trabalhar com eles está em: <https://www.youtube.com/watch?v=i_c5Fzk807M&list=PLHz_AreHm4dlUpEXkY1AyVLQGcpSgVF8s&index=17>

## Alinhamento e Formatação de Texto

Para alinhar o texto usamos a **propriedade** *text-align: ;*  com os determinados **valores:** *center, left, right, justify.* Há também **duas propriedade muito importante** para a formatação do texto, a *text-indent: ;* com ela é possível colocar um espaço no inicio de cada parágrafo (como apertar a tecla table no word) e line-height: ; que amenta o espaço entre linhas.

Exemplo:

body {

font: normal 16px Arial, Verdana, Serif;

text-align: left;

}

h1 {

text-align: center;

}

h2 {

text-align: right;

}

p {

text-align: justify;

text-indent: 50px;

line-height: 2em;

}

## Formatando Listas

Para colocar ícone da **lista dentro da caixa** usamos a declaração: list-style-position: inside; Para **dividir a lista em colunas** usamos a **propriedade** columns: ; com o **valor** de colunas. Para **alterar o ícone da lista** usamos a **propriedade** list-style-type: ; mas esta propriedade não funciona em todo navegador. Veja o exemplo:

ul {

    list-style-position: inside;

    columns: 2;

    list-style-type: '\2714\00A0\00A0';

}

(O código 2714 é de um símbolo pegado na emojipedia e 00A0 acrescenta espaços)

## Seletores Personalizados Id e Class

Para criar estilos personalizados, utilizamos os seletores de **id** **(#)** e de **class** **(.)** de maneira eficiente. Ao criar nosso conteúdo em HTML, podemos identificar um determinado elemento único com um id, ou agrupar elementos múltiplos que tenham características semelhantes com um class. Vamos olhar um exemplo simples de documento HTML, com propriedades identificadoras:

<body>

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

<h1 class="destaque">Aprendendo HTML</h1>

<h2 class="basico">HTML Básico</h2>

</body>

h1 {

color: blue;

}

#principal {

text-align: center;

background-color: rgb(255, 255, 255);

font-weight: bold;

}

.basico {

color: green;

}

.destaque {

background-color: rgb(231, 214, 114);

}

O seletor id deve ser usado apenas uma vez dentro de um único documento HTML enquanto o seletor class pode ser usado em vários elementos. É possível colocar varias classes em um mesmo elemento e também é possível usar o seletor id e class em um único elemento.

Em HTML é id = em CSS é #

Em HTML é class = em CSS é .

## 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.**

Já um **pseudo-elemento** CSS é uma palavra-chave adicionada às declarações de um seletor após dois sinais de dois pontos e permitem que você formate um pedaço específico do elemento referenciado. Os principais pseudo-elementos usados nas CSS são **::before, ::after, ::first-letter, ::first-line.**

Sinal de maior **>** significa filho-direto do elemento. Seleciona todos os filhos-diretos do elemento. Sendo assim podemos usar este sinal para fazer uma determinada ação com uma pseudo-classe a um elemento específico.

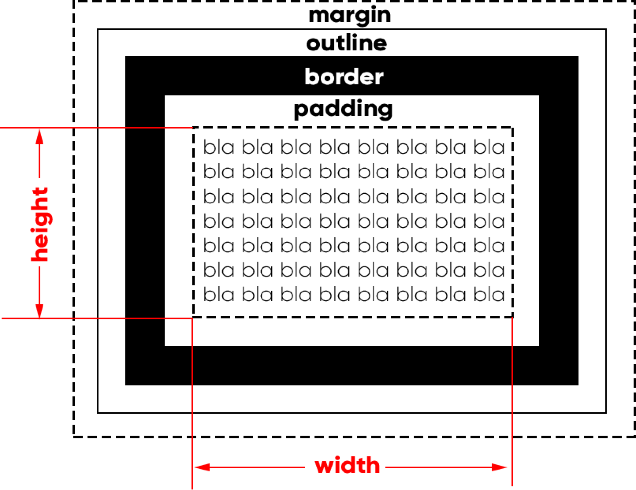
Para mais informações assista os vídeos das aulas 21 e 22 do segundo módulo de HTML e CSS no canal do Curso em Vídeo do YouTube:

<https://www.youtube.com/watch?v=WPtRX4n0UJs&list=PLHz_AreHm4dlUpEXkY1AyVLQGcpSgVF8s&index=21>

<https://www.youtube.com/watch?v=vMlrcOVr7po&list=PLHz_AreHm4dlUpEXkY1AyVLQGcpSgVF8s&index=22>

## Modelos de Caixas: Block-Level e Inline-Level

Vamos analisar como uma caixa vai ser apresentada por todos os navegadores. Olhe atentamente o diagrama a seguir, que é exatamente um modelo de caixa:



Dependendo do comportamento da caixa, podemos classificar um elemento em uma de duas categorias:

**Caixa do tipo block-level**

Um elemento dito **block-level** sempre vai se iniciar em uma nova linha e vai ocupar a largura total do elemento onde ele está contido. Se não estiver contido em nenhuma outra caixa, ele vai ocupar 100% da largura do <body>.

Exemplos:

<address> <article> <aside> <blockquote> <canvas> <dd> <div> <dl> <dt> <fieldset> <figcaption> <figure> <footer> <form> <h1> - <h6> <header> <hr> <li> <main> <nav> <noscript> <ol> <p> <pre> <section> <table> <tfoot> <ul> <video

**Caixa do tipo inline-level**

Um elemento do tipo **inline-level** não vai começar em uma nova linha, e sim no ponto exato onde foram definidos. E a largura dele vai ocupar apenas o tamanho relativo ao seu conteúdo.

Exemplos:

<a> <abbr> <acronym> <b> <bdo> <br> <button> <cite> <code> <dfn> <em> <i> <img> <input> <kbd> <label> <map> <object> <output> <q> <samp> <script> <select> <small> <span> <strong> <sub> <textarea> <tt> <var>

## Configurações de Caixa em CSS e ShortHand

Exemplo de formatação de caixa

  h1 {

            background-color: lightgray;

            height: 300px;

            width: 300px;

            border-width: 10px;

            border-style: solid;

            border-color: darkgrey;

            padding-top: 10px;

            padding-right: 10px;

            padding-bottom: 10px;

            padding-left: 10px;

            margin-top: 10px;

            margin-right: auto;

            margin-bottom: 40px;

            margin-left: auto;

            outline-width: 5px;

            outline-style: dashed;

            outline-color: blueviolet;

        }

        a {

            border-width: 5px;

            border-style: solid;

            border-color: red;

            padding-top: 10px;

            padding-right: 10px;

            padding-bottom: 10px;

            padding-left: 10px;

        }

    </style>

Há varias propriedades para formatação de caixa:

Height: configura altura da caixa (somente em block-level)

Width: configura largura de caixa (somente em block-level)

Border: configura borda em pixel, estilo e cor

Padding: configura a distância entre a caixa de texto e a borda

Margin: configura a parte externa da borda, usado para aumentar da distancia entre uma caixa e outra

Outline: configura uma linha externa da borda, respeitando o limite da margem

Vimos todas essas propriedades no exemplo anterior, mas também é possível usar **Shorthand** para simplificar os códigos.

Nos casos de **border e outline** a sequência é **width -> style -> color.**

Nos casos de **margin e padding** a sequência é o sentido horário **top -> right -> bottom -> left,** se preencher com apenas um valor, será gerado o mesmo valor para todos os lados, no caso de dois valores, o primeiro valor configura **top** e **bottom** enquanto o segundo valor configura **right** e **left.**

Veja no exemplo abaixo:

<style>

        h1 {

            background-color: lightgray;

            height: 300px;

            width: 300px;

            border: 10px solid darkgrey;

            padding: 10px;

            margin: 10px auto 40px auto;

            outline: 5px dashed blueviolet;

        }

        a {

            border: 5px solid red;

            padding: 10px 5px;

        }

    </style>

Há também um parâmetro interessante o display.

Display: pode receber os valores **block** e **inline** alterando a característica da caixa de block-level para inline-level ou vice e versa.

## Sombras em Caixas

Para criar sombras usamos a propriedade ***box-shadow: ;***

Ela também é uma **shorthand** e aceita quatro valores principais nesta ordem:

Deslocamento horizontal -> deslocamento -> espalhamento -> cor

Pode também fazer sombra interna na caixa com o valor **inset** antes do deslocamento horizontal.

Exemplo:

nav {

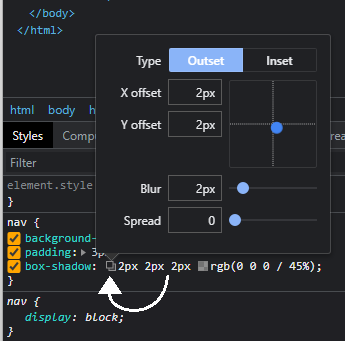
    background-color: rgb(100, 100, 100);

    padding: 3px;

    box-shadow: 2px 2px 2px rgba(0, 0, 0, 0.445);

}

No **DevTools** é possível ver um pequeno quadrado antes do primeiro valor, nele é possível alterar em tempo real e em seguida copiar o código gerado e colar no arquivo CSS.



## Bordas Arredondadas nas Caixas

Para criar sombras usamos a propriedade ***border-radius: ;***

Ela também é uma **shorthand** e aceita quatro valores principais nesta ordem:

top-left -> top-right -> bottom-right -> bottom-left. Mas **ao indicarmos um único** valor todas as bordas ficam arredondadas na mesma proporção. É possível colocar o valor em pixel e em porcentagem. Por exemplo se quisermos criar uma caixa redonda (um círculo) basta colocar a declaração *border-radius: 50%;*

Exemplo de borda arredondada:

main {

    background-color: white;

    padding: 10px;

    margin: 10px;

    border-radius: 10px;

}

No **DevTools** é possível alterar valor por valor, ver em tempo real e em seguida copiar o código gerado e colar no arquivo CSS.

## Montagem da Estrutura do Site

Para criar o modelo de caixas acesse o site: <https://mockflow.com/>

## Criando Variáveis em CSS

Para criar variáveis em CSS basta usar a pseudo-classe **:root** se equipara à raíz do documento. Aplicado ao HTML, **:root** representa o elemento <html> e é identico ao seletor html, exceto que sua especificidade é mais alta.

A partir do **:root** usamos **- -** para criar uma vairável.

Exemplo de variáveis que armazenam cores:

:root {

    --cor0: #c5ebd6;

    --cor1: #83e1ad;

    --cor2: #3ddc84;

    --cor3: #2fa866;

    --cor4: #1a5c37;

    --cor5: #063d1e;

}

Exemplos de variáveis que armazenam fonts:

:root {

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

    --fonte-destaque:'Bebas Neue', cursive;

    --fonte-android: 'idroid', cursive;

}

Para usar a variável criada basta usar normalmente a propriedade especifica da variável como nos casos acima, color e font-family. Elas vão estar no final da lista de valores, basta apertar Ctrl + espaço e descer até a final.

Exemplo de variável em uso:

body {

    background-color: var(--cor0);

    font-family: var(--fonte-padrao);

}

header {

    background-color: var(--cor5);

}

## Site Responsivo

Para criar um site responsivo primeiro temos que adaptar as **caixas do conteúdo a largura** **máxima e mínima**. Para isso usamos as propriedades min-width e max-width com vares em px. Também usamos a declaração margin: auto; se caso o site for acessado de telas muito grades a **caixa fica centralizada.** Outra coisa importante é especificar uma **altura min para o cabeçalho** com a propriedade min-height, isso faz com que mesmo em tela grande ele possa ter uma altura adequada.

Para **imagens responsivas** podemos usar a tag *picture* em HTML, veja mais a respeito no capítulo de imagens dinâmicas na pagina 8. E também podemos usar a declaração width: 100%; que **faz a imagem crescer e diminuir conforme a tela**. Para **centralizar uma imagem** usamos a declaração margin: auto; junto com a declaração display: block; isso porque uma imagem é do tipo inline-level . Para que a **imagem não cresça** muito a ponto de resolução ficar ruim deve se usar a propriedade max-width: ; e estipular um valor em px.

Para deixar um **vídeo do Youtube mais responsivo** temos que envelopa-lo em um tag *div* dentro do HTML e fazer as alterações a partir dela.

Veja o exemplo a baixo:

div.video {

    padding-bottom: 59%;

    position: relative;

}

div.video > iframe{

    position: absolute;

    top: 5%;

    left: 5%;

    width: 90%;

    height: 90%;

}

Dependendo do projeto os valores podem sofrer alterações.

## Planos de Fundo

Para criar plano de fundo com imagem usa-se a declaração backgroud-image: url(‘.....’). Dentro de url é possível colocar um caminho de um arquivo, ou o link de uma imagem online.

Exemplo 1:

 body {

            background-image: url('https://gustavoguanabara.github.io/html-css/imagens/mascote.png');

            background-size: 100px 100px; /\*Altera o tamanho\*/

            background-repeat: no-repeat; /\*Faz a imagem não repetir, podendo usar outros valores como: repeat-x e repeat-y\*/

        }

Exemplo 2:

body {

            height: 98vh; /\*98% da altura da viewport\*/

            background-image: url('imagens/wallpaper003.jpg');

            background-position: center center; /\*Esta propriedade se refere a posição de onde a imagem fica ancorada, o primeiro valor é coluna e o segunda é linha\*/

        }

Exemplo 3:

 body {

            height: 100vh;

            background-color: black;

            background-image: url('imagens/wallpaper002.jpg');

            background-position: center center;

            background-repeat: no-repeat;

            background-size: cover; /\*Faz com que a imagem seja mostrada inteira sem distorcer\*/

            background-attachment: fixed; /\*Deixa o plano de fundo fixo\*/

        }

**Para criar um plano de fundo fixo** que ocupe a **tela inteira** onde somente o conteúdo role para baixo usamos as seguintes declarações no exemplo 3:

background-size: cover; com background-attachment: fixed;

E podemos **simplificar o exemplo 3 com uma shorthand** background: ; onde a sequencia dos valores é: color > image > position > repeat > [size] > attachment. **Observação,** o **size está na regra mas ainda não está funcionando** na shorthand então devemos configurar ele separademente. Então conseguimos simplificar o exemplo 3 desta forma:

 body {

            height: 100vh;

            background: black url('imagens/wallpaper002.jpg') center center no-repeat fixed;

            background-size: cover;

        }

## Centralização de Conteúdo na Vertical

Para exemplificar criamos uma **tag article com id conteudo** dentro de uma **tag section com id container,** onde container terá a cor roxa e conteudo será menor e terá a cor amarela.

**Container** deve ser colocado em posição relativa e **conteudo** em posição absoluta.

Na sequencia podemos configurar a posição de **conteudo** com left 50% e top 50%, isso faz com que o topo esquerdo de **conteudo** fique centralizado em **container.**

Para finalizar basta colocar a declaração transform: translate(-50%, -50%); em **conteudo.**

Exemplo:

<style>

        #container {

            background-image: url('imagens/target001.png');

            background-size: 100% 100%;

            position: relative;

            height: 95vh;

            background-color: purple;

            padding: 10px;

        }

        #conteudo {

            background-image: url('imagens/target001.png');

            background-size: 100% 100%;

            position: absolute;

            height: 200px;

            width: 400px;

            background-color: yellow;

            left: 50%;

            top: 50%;

            transform: translate(-50%, -50%);

        }

    </style>

</head>

<body>

    <section id="container">

        <article id="conteudo">

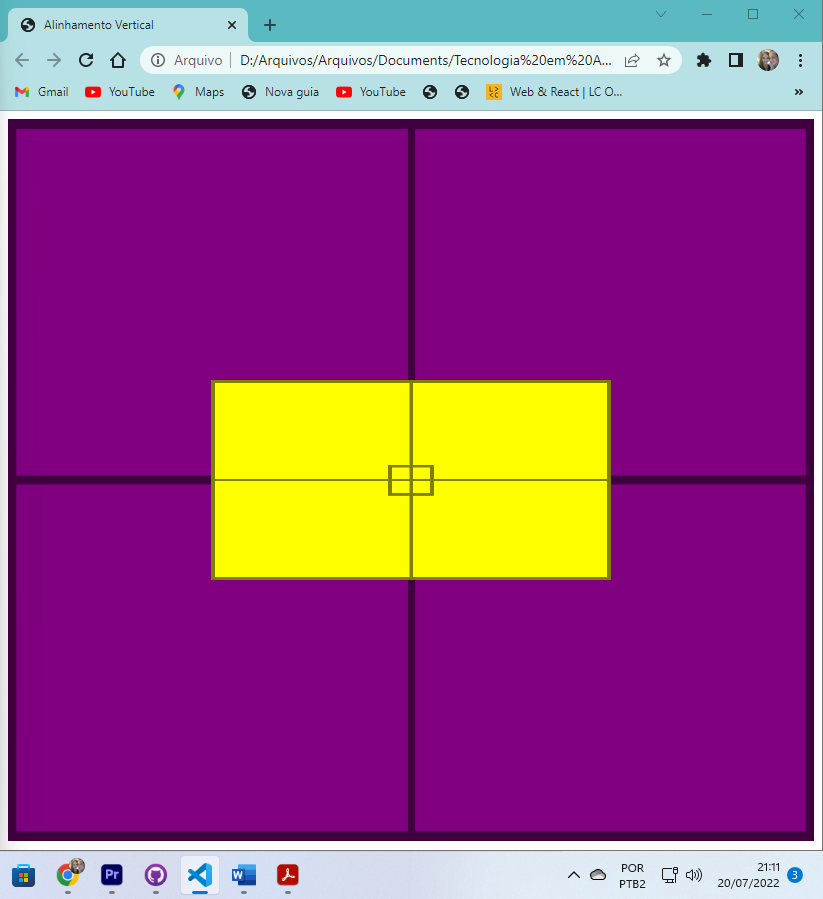
        </article>

    </section>

</body>

</html>

Veja o resultado na imagem:



# Tags HTML

*<!-- -->* : Cria comentários.

*<h1> </h1> ao <h6> </h6>:* Coloca títulos na página sendo que h1 é o primeiro da hierarquia de títulos.

*<p> </p>:* Cria parágrafos.

*<hr>:* Cria uma linha horizontal.

*<br>:* Quebra de linha.

*<img src=”” alt=””>:* Inseri uma imagem. Para não precisar ficar digitando o caminho e o nome da imagem basta clicar entre as aspas do src, segurar Ctrl e apertar espaço para achar o caminho do arquivo e entre as aspas do alt coloque uma descrição curta da imagem.

*<i> </i>:* O texto fica em itálico. **OBS: não semântico.**

*<em> </em>:* O texto fica em itálico e indica ênfase.

*<address> </address>:* O texto fica em itálico e indica que é um endereço.

*<b> </b>:* O texto fica em negrito. **OBS: não semântico.**

*<strong> </Strong>:* O texto fica em negrito e indica que ele está em destaque.

*<u> </u>:* O texto fica sublinhado. **OBS: não semântico.**

*<ins> </ins>:* O texto fica sublinhado e indica texto inserido.

*<del> </del>:* O texto fica riscado e indica texto excluído.

*<mark> </mark>:* Adiciona um marca-texto. *Obs: para mudar a cor de um marca-texto basta usar css dentro da tag.* ***Exemplo: <mark style=”background-color: lime;”>***

*<small> </small>:* Deixa o texto pequeno.

*<sup> </sup>:* Deixa o texto sobrescrito.

*<sub> </sub>:* Deixa o texto subscrito.

<*code> </code>:* Deixa as fontes monoespaçadas, ou seja, todas as letras ocupam o mesmo espaço.

*<pre> </pre>:* Formata todo texto conforme escrito no documento HTML respeitando até mesmo as quebras de linha.

*<q> </q>:* Coloca o texto entre aspas indicando uma citação.

*<blockquote> </blockquote>:* Indica que o texto é uma citação. *Obs: podendo colocar o link da citação na tag.* ***Exemplo: <blockquote cite=”http//:www........”> Bla bla bla <blockquote>.***

*<abbr> </abbr>:* Usado para mostrar o significado de uma abreviação quando passar o mouse por cima dela.

*<ol> </ol>:* Cria listas ordenadas.

*<ul> </ul>:* Cria listas não ordenadas.

*<li>:* Cria as linhas de uma lista ordenada ou não ordenada.

*<dl> </dl>:* Cria lista de definições.

*<dt> </dt>:* Cria um termo para uma lista de definições.

*<dd> </dd>:* Cria a definição de um termo de uma lista de definições.

*<a href="........"* *> </a>:* Cria um link.

*<Picture> </Picture>:* Cria vários sources de imagens ao mesmo tempo.

*<audio src=””> </audio>:* Adiciona um áudio.

*<vídeo src=””> </vídeo>:* Adiciona um vídeo**. OBS. Essa opção gera custos autos de hospedagem. Veja em anotações.**

<span> </span>: é um container genérico em linha para conteúdo fraseado. Ele pode ser usado para agrupar elementos para fins de estilo (usando os atributos class ou id ), ou para compartilhar valores de atributos, como lang. <span> é muito parecido com o elemento <div>, entretando <div> é um elemento de nível de bloco enquanto <span> é um elemento em linha.

<div> </div>: é um container genérico para conteúdo de fluxo. Ele pode ser utilizado para agrupar elementos para fins de estilos (usando class ou id), ou para compartilhar valores de atributos, como lang. Ele deve ser utilizado somente quando não tiver outro elemento de semântica (tal como <article> ou <nav>).

# Entendendo o CSS

**Regra:** Uma regra CSS é uma declaração que segue uma sintaxe própria e que define como será aplicado estilo a um ou mais elementos HTML. Um conjunto de regras CSS formam uma Folha de Estilos.

**Seletores:** Um seletor CSS é a primeira parte de uma regra CSS. É um padrão de elementos e outros termos que informam ao navegador quais elementos HTML devem ser selecionados para que os valores de propriedade CSS dentro da regra sejam aplicados a eles.

**Propriedade:** é o atributo do elemento HTML ao qual será aplicada a regra (por exemplo: font, color, background, etc...).

**Valor:** é a característica específica a ser assumida pela propriedade (por exemplo: letra tipo arial, cor azul, fundo verde, etc...)

**Declarações:** É o conjunto entre propriedade e valor. Qualquer motor do CSS calcula quais declarações serão aplicadas para todos um único elemento da página em ordem adequadamente, a fim de exibí-lo com o estilo correto.

**@:** São regras que dever ser sempre colocadas nas primeiras linhas do arquivo CSS ou da tag style e não deve ser colocada dentro de seletores.

Exemplo:

**@import** url('https://fonts.googleapis.com/css2?family=Work+Sans:ital,wght@0,100;0,200;0,300;0,400;0,500;0,600;0,700;0,800;0,900;1,100;1,200;1,300;1,400;1,500;1,600;1,700;1,800;1,900&display=swap');

**body {**

**font-family: 'Work Sans', sans-serif;**

**font-weight: bolder;**

**font-size: 3em;**

**font-style: italic;**

**}**

# Programas e Extensões do Chrome

## Programas

**Visual Studio Code:** (VS Code) é um editor de código aberto desenvolvido pela Microsoft.

**Gimp:** É um programa de código aberto voltado, essencialmente, para edição e criação de imagens raster e, em menor escala, para desenho vetorial. O software é repleto de recursos, de fácil uso e uma boa alternativa gratuita ao mais conhecido dos editores, o gigante Adobe Photoshop.

**HandBrake:** É um programa multiplataforma e multitarefa em código aberto de conversão de arquivos de vídeo e DVD e Bluray para MPEG-4, disponível para Mac OS X, Linux e Windows. Com ele é possível compactar bastante um vídeo.

**Git:** Programa para versionamento local.

**GitHub Desktop:** Programa para versionamento online direto no site do GitHub.

## Extensões do Chrome

**Todas as extensões devem ser baixadas no site**: <https://chrome.google.com/webstore/category/extensions?hl=pt-br>

**Web Developer:** Essa extensão foi criada para concentrar diversas ferramentas simples e úteis em uma barra adicional no Chrome. O conceito veio da barra de ferramentas do desenvolvedor.Ela é repleta de utilidades para quem quiser usar em seu trabalho diário, não só para os programadores, mas também para quem opera no webdesign. Dentre os recursos estão alguns como:adição de contornos aos elementos (tudo com apenas um clique);detecção de imagens quebradas;controle de habilitação/desabilitação de popups, plugins e JavaScript.

**ColorZilla:** É uma extensão de navegador muito versátil que permite descobrir um código de cor específico para um determinado site. Além disso, esta ferramenta acaba sendo indispensável para designers gráficos, desenvolvedores web ou front-end.

**Fonts Ninja**: Identifica fontes de qualquer site, que não esteja dentro de uma imagem.

**Window Resizer:** permite emular diversas resoluções de ecrã para testar as suas páginas de Internet. Após instalada é necessário fazer uma permissão, para isso abra o menu de extensões, gerencias extensões e clique em saiba mais do ícone do Window Resizer e ligue “permitir acesso a URLs de arquivo”.