# **Curso HTML/CSS Módulo 1**

Professor Gustavo Guanabara

No HTML5, tomar cuidado com a **semântica**, ou seja, o significado das **tags**. Semântica é muito importante, pois o próprio Google dá preferência na busca para sites com semântica. Existem tags que ainda funcionam, mas só são para estilo, e hoje, estilo é totalmente com CSS. Sempre preferir tags com semântica.

## **Cabeçalho/Head**

**Meta**: acentuação e tamanho do viewport (apenas no head);

**Title**: título da página que vai aparecer no navegador.

Texto

Descrição gerada automaticamente

## **Inserindo comentários**

<!-- Só de colocar isso no VS, ele já adiciona a tag do comentário, que completa é **<!-- texto -->**.

## **Usando uma tag como texto**

Se quisermos mencionar a frase “p e /p”, para que não vire uma tag, devemos colocar assim: **&lt;p&gt e &lt;/p&gt**. Isso significa less than (menor que) e greater than (maior que).

## **Símbolos especiais**

Conseguimos adicionar diversos símbolos no html, basta digitar **&** e o código do símbolo. Alguns exemplos:

Tabela

Descrição gerada automaticamente

## **Emojis**

Ver em [www.emojipedia.org](http://www.emojipedia.org), buscar o emoji e embaixo o código do que queremos inserir. Sempre colocar inicialmente **&#x** e o código que tá lá no site. Exemplo: **&#x1F436** (cachorrinho).

## **Favicon**

Pequena imagem do site que o identifica mais facilmente no navegador. Formato ICO. Ir no site <https://favicon.io/> ou <https://www.iconarchive.com/> e baixar de lá, colocar na mesma pasta. Para inserir, ir na parte do **head** do código, acima do title, digitar link e escolher a opção favicon. Mudar **o** href, colocar o caminho do favicon:



## **Negrito**

A tag **<strong>** significa que o termo delimitado possui força dentro da frase. Logo, ele aparecerá em negrito. A tag <b> também funciona, mas não tem semântica, é apenas uma marca de forma, não está informando ao html que é um texto importante;

## **Itálico**

A tag **<em>** significa que queremos dar ênfase ao termo. Logo, ele aparecerá em itálico. A tag **<i>** também funciona, mas não tem semântica, é apenas uma marca de forma, não está informando ao html que é um texto importante;

## **Ênfase a endereços**

A tag **<adress>** significa que queremos dar ênfase a um endereço, e estamos informando ao navegador que aquilo é um endereço. No caso, aparecerá em itálico, e talvez até conseguiremos abrir no Google Maps em outras plataformas;

## **Texto sublinhado**

A tag **<mark>** deixa como se fosse com caneta marca texto;

**Dica de envelopamento de um texto**

Ctrl shift P: ir em “Emmet: wrap with abbreviation”, dar enter, e depois colocar a tag que queremos, tipo Strong. Muito mais fácil quando esquecemos de colocar uma frase numa tag.

## **Texto menor**

A tag **<small>** deixa o texto menor. A tag <**big>** deixava o texto maior, mas parou de existir;

## **Texto riscado**

A tag **<del>** deixa o texto com o risco ao meio;

## **Texto sublinhado**

A tag **<ins>** deixa o texto com o risco ao meio. A tag **<u>** também funciona, mas não tem semântica, é apenas uma marca de forma, não está informando ao html que é um texto importante;

## **Texto sobrescrito**

A tag **<sub>** é usada para deixar acima, tipo x2.

## **Texto subscrito**

A tag **<sup>** é usada para deixar acima, tipo h2o.

## **Destacando códigos**

A tag **<code>** deixa o texto com uma fonte monoespaçada, que é aquela utilizada em códigos e que tem o mesmo tamanho para cada letra. A tag **<pre>** serve para deixar o código com os espaçamentos à esquerda, bem certinho.

## **Citações simples**

A tag **<q>** deixa o texto entre aspas, de forma semântica (pois até daria para colocar as aspas manualmente, mas nesse caso é uma fala ou citação de terceiros). É uma citação simples, na mesma linha.

## **Citações completas**

A tag **<blockquote>** deixa o texto um recuo à esquerda, como se fosse um parágrafo de um livro. Já a tag **<cite>** serve para citar quem falou ou quem é o dono de tal citação, não há diferença nenhuma visualmente, mas é semântica e faz com que os mecanismos de busca identifiquem que essa frase é de tal livro ou pessoa;

## **Abreviações**

A tag **<abbr>** deixa um pontilhado embaixo do texto, e ela abre de forma que a gente consiga informar no **title** o que significa aquela sigla quando encosta o mouse:



Texto

Descrição gerada automaticamente

## **Texto invertido**

A tag **<bdo>** deixa o de forma invertida. Há 2 tipos: **bdo l (left),** que vai deixar o texto de left to right; e a **bdo r (right),** que vai deixar o texto de right to left. A opção **bdo r** deixa o texto invertido, bdo l não faz nada:



## **Listas**

### **Listas ordenadas**

A tag **<ol>** faz uma ordered lists (ol), que são aquelas listas onde a ordem dos itens é algo muito importante. A tag **<li>** (list item)serve para identificar cada item. Dá para configurar o marcador da lista com a tag **<type>**,ou seja, informar se é uma lista numérica **(1)**, algarismo romano maiusculo **(I),** algarismo romano minúsculo **(i),** lista com letra maiúscula **(A**) ou letra minúscula **(a).** Dá para usar também a tag **<start>,** informando de onde a lista vai começar:

Texto

Descrição gerada automaticamente Texto

Descrição gerada automaticamente

### **Listas não ordenadas**

A tag **<ul>** faz uma unordered lists (ol), também chamadas de listas com marcadores, que são aquelas onde a ordem dos itens não influenciará no significado da lista. A tag **<li>** (list item)serve para identificar cada item. Dá para configurar o marcador da lista com a tag **<type>**,ou seja, informar se a lista deve ser marcada com círculo pintado **(disc),** círculo vazado **(circle)** ou quadrado **(square)**:

Texto

Descrição gerada automaticamente Texto

Descrição gerada automaticamente

### **Juntando listas**

Dá para juntar várias listas:

Texto

Descrição gerada automaticamente Texto, Carta

Descrição gerada automaticamente

### **Listas de definição**

A tag **<dl>** faz uma definition list, que basicamente é um dicionário, onde temos os termos e as descrições. A tag **<dt>** (definition term)representa cada termo, e a **<dd>** (definition description) é cada descrição. Dá para configurar o marcador da lista com a tag **<type>**,ou seja, informar se a lista deve ser marcada com círculo pintado **(disc),** círculo vazado **(circle)** ou quadrado **(square)**:

Texto

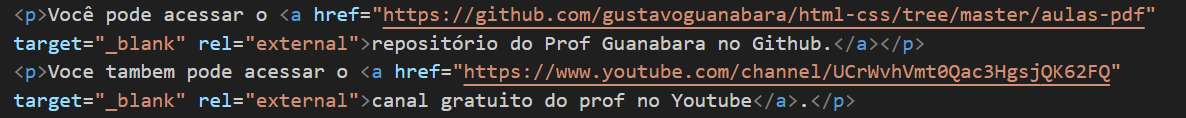
Descrição gerada automaticamente**Interface gráfica do usuário, Texto

Descrição gerada automaticamente**

## **Links**

### **Link externo (site de fora)**

A tag **<a>** faz um link com um site externo, ou seja, uma url., que basicamente é um dicionário, onde temos os termos e as descrições. O atributo **<href>** serve para colocar o próprio link; já o **<target>,** opcional,mas importante, serve para que o navegador abra uma nova janela **(\_blank),** ou na própria página **(\_self)**. O atributo **<rel> serve** para demonstrar se o site é externo (**external**), próxima parte de um site (next), parte anterior do site (**prev**), para um site do autor do próprio site (**author**), ou quando precisamos colocar um link mas não sabemos se é um link oficial, confiável ou não queremos dar um aval para aquele contéudo **(nofollow)**:

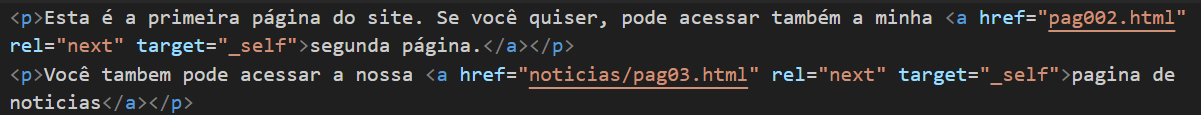


### **Para criar um link**

Apertar Ctrl e clicar em cima do link que colocamos!

### **Link interno (outra página dentro do site)**

Mesmas tags acima, mas no atributo **href** coloca-se o nome da página que foi criada (tipo index2.html, devendo estar na mesma pasta), no **target** vai ser o **\_self,** e no **rel** vai ser **next:**

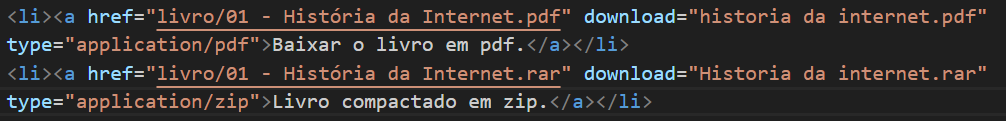


Se tiver uma terceira página de um site mas que esteja em outra pasta, se quiser colocar a referência para a primeira, devemos informar que tem que acessar uma “pasta acima” através de **../ :**



### **Link para downloads**

Mesmas tags acima, mas no atributo **href** coloca-se o nome do arquivo para fazer o download. Um novo atributo é o **download,** queé o nome do arquivo, e o atributo **type,** que no caso de um pdf, devemos colocar “application/pdf”. Para saber o que escrever dentro do **type,** acessar<https://www.iana.org/assignments/media-types/media-types.xhtml>:



Aqui vão alguns media types bem usados no nosso dia-a-dia:

* application/zip
* text/html
* text/css
* text/javascript
* video/mp4
* video/H264
* video/JPEG
* audio/aac
* audio/mpeg
* font/ttf
* image/jpeg
* image/png

## **Imagens**

Não é correto utilizar imagens da internet nos nossos sites, principalmente profissionalmente. Devemos contratar um fotografo ou ilustrador, tirar a própria foto, ou até contratar uma imagem nos bancos de imagens, como iStockPhoto, DepositPhotos, ShutterStock, etc. Mas no google, conseguimos ter acesso à imagens gratuitas procurando assim: Google imagens > Ferramentas > Marcar **“Licenças Creative Commons”.** Não ir emLicenças comerciais ou outras, pois essas imagens têm licenças que não são Creative Commons e podem ser de sites disponíveis sem custos financeiros ou de sites comerciais que exigem pagamento. Abaixo alguns sites para pegar imagens gratuitamente:

• UnSplash

• Pexels

• FreePik

• Rawpixel

• Pixabay

• Libreshot

• Wikimedia Commons

Basicamente existem 2 tipos principais de imagens para web, e devemos nos atentar quando utilizar cada um:

* **JPEG (criado em 1983, gerenciado por Joint Photographics Experts Group):** formato com imagens extremamente leves e compactos, o que é muito bom quando estamos fazendo um site que sempre deve ser leve e rápido, mas tem que tomar cuidado na hora de configurar o tamanho da compactação para que não fique horrível e toda borrada, pois a qualidade não é tão boa. Se quisermos colocar uma foto gigante no fundo da tela, usar esse formato;
* **PNG (criado em 1996, gerenciado por Portable Network Graphics)**: muito bom pois permite transparência, é leve, mas não tanto quanto o JPEG; quando quisermos uma imagem flutuando na tela com fundo transparente, é esse que vamos utilizar.

De forma resumida, na hora de escolher o formato de imagem para o seu site, opte sempre pelo formato **JPEG** com uma compactação entre 30% e 50%. O formato PNG só deverá ser usado quando precisarmos de transparência na foto.

Sobre o **tamanho das imagens**, a regra de ouro nesses casos é: use imagens do tamanho certo! Se precisar de uma imagem com 200 pixels de largura, gere um arquivo exatamente com esse tamanho. **Não ficar salvando arquivos gigantes** e diminuindo o tamanho da imagem com códigos, até porque não vai ficar mais leve se mudar a largura dela no seu CSS, o contrário, o navegador vai levar um tempão para carregar o arquivo do servidor e vai exibir a imagem minúscula na tela.

Para adicionar imagens, usar a tag **<img>.** O primeiro será onde a imagem está **(src=source, origem)**, que pode estar junto com o arquivo index, ou em outra pasta, ou mesmo num navegador. Depois, é o texto alternativo da imagem **(alt=alternative, alternativo),** que serve para quando um visitante for deficiente visual, o navegador vai ser capaz de descrever que tipo de foto está sendo mostrada ali. É importante para a acessibilidade.

Usar ctrl + espaço para achar o local da imagem:

**Texto

Descrição gerada automaticamente**

## **Imagens dinâmicas (que se adaptam)**

São imagens que se adaptam de acordo com o tamanho pré-definido da view port (janela). Por exemplo, temos 3 imagens: uma de 1000px x 1000px (G); outra de 700px x 700px (M); e outra de 300px x 300px (P).

Inserir a tag **<picture>** (indica que vai vim uma imagem logo a seguir) e, dentro dela, **<img>,** inserindoaimagem G, que é a imagem base que se quer carregar. Antes da tag img, inserir a tag **source** **mediatype,** que serve para informar alguns parâmetros de tamanho. Essa tag funciona assim: dentro dela, temos o atributo **media,** que serve para dizer o tamanho máximo ou mínimo da imagem; o outro atributo **srcset,** vamos colocar as imagens secundárias, ou seja, as imagens que vai carregar se os parâmetros não atenderem para carregar a imagem G. Por último, atributo **type,** informando o tipo da imagem (jpeg ou png):

Tela de computador com letras brancas

Descrição gerada automaticamente

Resumindo: até 750 px, vai carregar a imagem P; se passar disso e até 1050px, carrega a M; se passar, carrega a G e mantém nela. Tem que seguir a sequência, no caso a P primeiro, que é a menor, até a G, a maior.

Isso é muito útil para a UX experience, pois o site fica mais leve e mais bonito, não fica aquela imagem enorme se o usuário abrir no celular por exemplo.

## **Áudios**

Áudios gratuitos**: Ir no YouTube > Perfil > YouTube Studio > do lado esquerdo, Biblioteca de áudio.** Dá pra fazer o download!

Para adicionar um áudio no site, usar a tag <**audio>.** Informar o atributo **src**, que é a fonte, ou local onde está, e colocar o atributo **controls autoplay:**

****

Outra forma de inserir áudios de um jeito que não dê problema em nenhum navegador é: inserir normalmente a tag **<audio>,** e dentro dele colocar o atributo **source**. No source, colocar o **src,** que é onde o arquivo está, e o atributo **type,** que é o tipo de áudio (pode ser mpeg, ogg, wav). Sendo assim, o que estiver primeiro é o que o navegador vai tentar reproduzir; se não conseguir, vai para o próximo, e assim por diante. O atributo **preload** da tag **áudio** é bem importante, pois ele que vai indicar se o áudio deve ser carregado ou não. Se informar **auto**, o site vai carregar o áudio inteiro assim que abrir a página (bem perigoso, pois vai ficar pesado); se colocar **metadata**, vai carregar só algumas informações do áudio, e tem o **none,** que não vai carregar nada. Temos também o atributo **loop,** que recomeça o áudio quando ele chega no final:

Texto

Descrição gerada automaticamente

## **Vídeos**

**Dica de programa para fazer edição de vídeos**

**HandBrake**, baixado na máquina

### **Vídeos em hospedagem própria (localmente na própria máquina)**

Temos diversos formatos, dentre eles, o **m4v** (mais recomendado), **mp4, ogg e webm**. Aqui, o ideal é que se tenha vários formatos para inserir todos, igual fizemos com as imagens dinâmicas e com os áudios, para se caso o navegador não suportar um, vai pro outro.

A tag para inserir vídeo é **<video>.** Existem vários atributos, dentre eles, o **width**, que vai indicar a largura que o vídeo vai ter na tela. O atributo **poster** configura uma imagem que vai aparecer como uma capa, enquanto o usuário não aperta o play (capa do vídeo). O atributo **controls** vai configurar se os controles do vídeo vão aparecer na parte inferior da mídia. Por padrão, os controles não aparecerão, mas basta colocar a palavra **controls** na tag **<video>.** O atributo **autoplay** diz para o navegador se o vídeo vai começar a tocar automaticamente.

Texto

Descrição gerada automaticamente

O problema de colocar assim, localmente, é que o tráfego de dados fica absurdo de caro para o cliente, pois está no servidor local. Vamos ver outras formas muito melhores e mais baratas.

### **Vídeos com hospedagem no YouTube**

Para pegar um site do **YouTube**, existe a opção de incorporar o vídeo no nosso código. Temos que faze esse passo a passo: Interface gráfica do usuário, Texto, Aplicativo, Email

Descrição gerada automaticamente

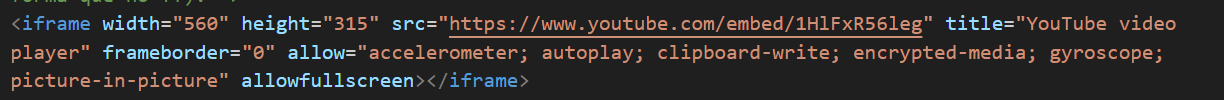
Interface gráfica do usuário, Texto, Aplicativo, Email

Descrição gerada automaticamente

Interface gráfica do usuário, Texto, Aplicativo

Descrição gerada automaticamente

Só colar esse código com a tag **<iframe>** no html, mas dá pra mudar algumas coisas também, tais como largura e altura:



Nesse caso, não consome os dados do servidor, e sim do YouTube. De grande valia, pois se o cliente tiver uma conta no Yt que é de graça e colocar os vídeos dele lá, dá pra usar só a tag **iframe** mesmo.

### **Vídeos com hospedagem no Vimeo**

Bom porque dá para deixar os vídeos privados, por exemplo, num site de assinatura de cursos (estudonauta), o usuário só consegue chegar até o vídeo se assinar. Para incorporar, fazer o passo a passo abaixo:

Para vídeos do próprio usuário:



Para vídeos de outras pessoas:

Interface gráfica do usuário, Aplicativo

Descrição gerada automaticamente

## **Tipos de CSS**

**CSS** quer dizer, *Cascading Style Sheet*, ou **folha de estilo em cascata,** e serve para formatar visualmente a página, seja em cores, fontes, espaços, formas, etc. Dá pra inserir de 3 formas, sendo a última a mais recomendada:

### **CSS inline**

A **CSS** **inline** deve ser inserida linha por linha, funciona, mas não é recomendável, além de deixar o código “poluído”, pois não há separação de html e css. Deve inserir a tag **<style>** em cada linha, e depois as declarações de estilo:

Texto

Descrição gerada automaticamente com confiança média

### **CSS interna/local**

Diferente do inline, a **interna/local** fica separada do html, fica na tag **head.** É funcional, mas só é recomendada se tiver apenas 1 página no site. Deve inserir a tag **<style> debaixo do title do head,** e tudo que for colocada dentro dela, que serão os seletores (h1, p, h2, etc), através de declarações (cada linha), vai ser estilo:

Texto

Descrição gerada automaticamente

### **CSS externa**

Também tem seletores (h1, h2, p) e declarações (background color, font, font-size) igual o interno, mas o **CSS externa** é o melhor, pois o estilo fica totalmente separado do html. Deve inserir o link da css **debaixo do title do head:**

Texto

Descrição gerada automaticamente

A página CSS, a parte, será assim (colocar o @charset “UTF-8”, sempre, para não dar problema com acentos):

Texto

Descrição gerada automaticamente

# **Curso HTML/CSS Módulo 2**

## **Representação das cores**

Existem 3 formas de representar cores por CSS:

### **Representação por nome**

Blue, white, yellow, red.. é muito simples, indicado apenas se for um site simples, ou mesmo para estudo:

Interface gráfica do usuário, Texto, Aplicativo

Descrição gerada automaticamente

### **Representação por código hexadecimal (rgb)**

Lembrar que **hexadecimal** quer dizer base hexa, ou seja, os algarismos são **0, 1, 2, 3, 4, 5, 6, 7, 8, 9, A, B, C, D, E, F.** São utilizados 6 dígitos que representam 65 milhões de cores.

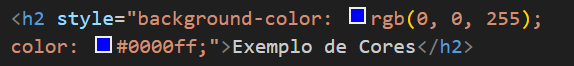
Todas as cores são derivadas de **vermelho (R), verde (G), e azul (B).** A representação fica assim: os 2 primeiros para vermelho (R), os 2 do meio verde (G), e os 2 últimos azul (B), formando o famoso **RGB**. Sendo que f = tudo da cor, e 0 = nada da cor. Cor preta é 000000, e branco ffffff. Cor vermelha pura é ff0000; cor verde pura é 00ff00; e cor azul pura é 0000ff.

Cada cor pode ir de 0 à 255 na escala:

Interface gráfica do usuário, Aplicativo

Descrição gerada automaticamente

Na CSS, pode colocar de 2 formas o RGB:



Informando: **rgb (0, 0, 255)** ou **#0000ff.** Dá para colocar transparência na cor também, aí adiciona a letra a e um dígito a mais (rgba).

### **Representação por características das cores (hsl)**

H = heel (matiz, cor) / S = saturation (saturação) / L = luminosity (luminosidade).

Interface gráfica do usuário, Texto

Descrição gerada automaticamente

Heel vai de 0 à 255; Saturation vai de 0 a 100%; e Luminosity de 0 à 100%. Também vira hsla se colocar transparência.

## **Tipos de cores e combinações**

Existe o **círculo cromático,** que serve para harmonizar as cores. Temos várias divisões de cores e coisas a observar:

**Cores primárias:** amarelo, azul e vermelho. Importante para gerar outras cores;

**Cores secundárias:** laranja, violeta e verde. Geradas pelas cores primárias;

**Cores terciárias:** cor primária + cor secundária. Ex: amarelo-esverdeado (sempre o nome da primária primeiro):

Gráfico, Gráfico de explosão solar

Descrição gerada automaticamente

**Temperatura das cores:** frias e quentes:

Gráfico, Gráfico de explosão solar

Descrição gerada automaticamente

**Cores complementares:** maior contraste, cor oposta no círculo. Serve para ser utilizado em fundo e cores de fonte, para aparecer bastante:

Gráfico, Gráfico de explosão solar

Descrição gerada automaticamente

**Cores análogas:** 3 cores vizinhas, irmãs, não tem contraste tão grande, ótimo para criar paletas:

Ícone

Descrição gerada automaticamente

**Cores análogas e uma complementar:** ótimo para criar uma paleta, pois sempre precisamos de uma cor com bastante contraste. 3 cores análogas, e 1 complementar:

Gráfico, Gráfico de explosão solar

Descrição gerada automaticamente

**Cores análogas relacionadas**: ótimo para criar uma paleta também sem tanto contraste. 2 cores análogas, pula uma, pega a próxima:

Gráfico, Gráfico de explosão solar

Descrição gerada automaticamente

**Cores intercaladas**: 3 cores, escolhe 1 cor, pula uma, pega a próxima, pula, pega a próxima novamente. Fica um pouco dura, Guana não indicou muito:

Gráfico, Gráfico de explosão solar

Descrição gerada automaticamente

**Cores triádicas:** 3 cores,outra forma de intercalar, mas aqui pula-se de 3 em 3:

Gráfico, Gráfico de explosão solar

Descrição gerada automaticamente

**Cores em quadrado:** 4 cores,outra forma de intercalar, mas aqui pula-se de 2 em 2:

Gráfico, Gráfico de explosão solar

Descrição gerada automaticamente

**Cores tetráticas:** 4 cores, mas deve-se formar um retângulo, pegando sempre a complementar:

Gráfico, Gráfico de explosão solar

Descrição gerada automaticamente

**Monocromia:** 1 cor só, apenas mexendo na saturação e luminosidade. É um perfeito degradê:

Gráfico, Gráfico de explosão solar

Descrição gerada automaticamente

## **Criando paletas de de cores**

O site <https://color.adobe.com/pt/create/color-wheel> é gratuito e ajuda muito a criar paletas.

Dá para criar todos os tipos de cores lá, e extrair tema de imagens (só arrastar pra lá), gradientes. Na aba “**Explorar**” e “**Tendências**”, aparecem milhares de sugestões de combinação de paletas.

O site <https://paletton.com/#uid=1000u0kllllaFw0g0qFqFg0w0aF> também é muito bom, dá pra ver até um pré visualização de um site na aba “Exemplo”.

O site <https://coolors.co/dde392-afbe8f-7d8570-646f58-504b3a> também é ótimo para inspirações, basta apertar espaço e ele vai gerando diversas paletas.

## **Capturando cores**

Ferramenta **Colorzilla** no google Chrome, ele consegue identificar qualquer cor da página ou imagem.

## **Configurações globais em CSS**

É necessário colocar algumas configurações antes de iniciar um trabalho de CSS. Para aplicar

Desenho em preto e branco

Descrição gerada automaticamente com confiança baixaNecessário para o gradiente pegar em toda a **altura** da tela.

## **Gradientes em CSS**

Temos que usar a propriedade **background-image:**

Interface gráfica do usuário, Texto

Descrição gerada automaticamente

Primeiro informar o tipo de gradiente, que pode ser **linear-gradient**, que deixa faixas, ou **radial-gradient**, que deixa em forma de círculo.

Depois, colocar o sentido do gradiente: **to top, 45deg, 180deg, to left** para **linear,** e **circle** para **radial**.

O % depois da cor indica o quanto da cor vai aparecer.