

# DEGRADE

background-image: linear-gradient(); - vai fazer com que o fundo do site fique em degrade

(background-image, linear-gradient)

dentro do parenteses: primeiro vc vai botar a direção e depois quantas cores quiser

---

background-image: linear-gradient(direção, quantas cores quiser);

direção - de que lado o degrade vai começar

---

background-image: linear-gradient(to right, white, blue);

vc pode indicar o lado usando a direção

background-image: linear-gradient(90deg, white, blue);

ou pode indicar o lado usando graus

---

\* {

} - (seletor especial) configurações globais das CSSs

---

altura

\* {

height: ; - muda a altura do site

height: 100%; - muda a altura do site em cem por cento

}

---

background-attachment: fixed; - deixa o fundo fixo (mesmo que role a tela vai ficar fixa)

---

background-image: radial-gradient() - outra forma de fazer degrade

background-image: radial-gradient(forma, quantas cores quiser)

---

background-image: radial-gradient(circle, blue, red)

---

---

# TIPOGRAFIA

## ANATOMIA DO TIPO

### Itens Metricos

Toda fonte quando é desenvolvida precisa se basear nos tamanhos, tamanho (espaço) das letras maiusculas e minusculas, tamanho vertical, tamanho horizontal, quanto espaço vai ocupar certa letra.

A letra x é muito importante, ela é o ponto de partida pra construção de todas as fontes.

Existe uma metrica chamada altura x, que é basicamente qual é a altura do x minuscuro na sua fonte, ela vai definir qual é a altura base de todas as letras minusculas sem contar os ascendentes e os descendentes.

Tambem existe a altura das letras maiusculas, chamada de altura das maiusculas

outra altura é esse vazamento que as letras minusculas tem, que as vezes é igual a altura das maiusculas em varias fontes.

Essa altura do vazamento pra cima se chama ascendente, que é o tamanho só do que vazou pra cima

Existe tambem a altura do que vazou pra baixo que se chama descendente

Todas as alturas juntas é chamada de corpo

Então quando vamos definir uma fonte nós temos que saber a altura do corpo dela, pq é a altura do corpo que configura nas CSSs

tudo isso que acabamos de falar são os chamados itens metricos, são aqueles que precisamos medir

todos os outros são os itens anatomicos, anatomicos geometricos e anatomicos

corporais.

---

Um dos elementos mais importantes da fonte, que é o pezinho da fonte, existe apenas em algumas fontes (A ← tem) (A ← não tem)

A partir de agora nem todas as fontes tem todos esses componentes que serão explicados.

esse pezinho ou prolongamento é chamado de serifa, essa serifa é um tracinho que gera no final de algumas fontes

A importância da serifa é que ela cria uma linha imaginária que nosso cérebro percebe, isso facilita a leitura das palavras, guiando a sua leitura

---

## Anatomicos Geometricos

A parte reta do k se chama haste, as partes curvadas do A se chama haste também

O que liga as duas hastes do A tem o nome de filete, o e também tem um filete

um filete une duas hastes ou um ou mais hastes e arcos

o que liga duas hastes e tem uma barriginha se chama arco

o n tem um arco

o b tem um saltinho em baixo, que serve pra criar a base da letra, que chamamos de esporão

a letra A ou a letra V tem o vertice, no A esta em cima e no V em baixo

quando temos um elemento e ele não esta preso em nenhum lugar o chamamos de terminal

---

## Anatomicos Corporais



---

## Glifos/Fonte

Glifo é uma letra, esse A é um glifo da fonte arial

cada uma das letras de um alfabeto é um glifo, quando pegamos o conjunto de todos esses glifos, em uma determinada família chamamos isso de fonte, fonte também pode ser o arquivo que você baixa para poder utilizar essa letra no computador

glifos - caracteres ou letras

fonte - conjunto de glifos

## Família tipográfica

Algumas fontes além de somente ter os glifos, têm também várias formas de representar o mesmo glifo. Isso é chamado de família tipográfica, nem toda fonte tem família tipográfica.

## Categorias de fontes

quando for escolher a categoria, existem várias.

serifadas: tem serifa

sans-serif: sem serifa

monoespaçadas: todas as letras/caracteres/glifos vão ter o mesmo tamanho (largura).

Toda fonte monoespaçada pode ser com ou sem serifa

handwriting (fonte scriptada): tenta simular a escritura a mão, simula a caligrafia humana

display (fonte comemorativa): não se preocupa em obedecer aquelas regras anatomicas

---

## Fontes em CSS

Para ver fontes e suas combinações seguras (safe fonts):

[https://www.w3schools.com/cssref/css\\_fonts\\_fallbacks.asp](https://www.w3schools.com/cssref/css_fonts_fallbacks.asp)

[https://www.w3bai.com/en-US/cssref/css\\_websafe\\_fonts.html?msclkid=bba5a462b0de11ec818276f93e804f77](https://www.w3bai.com/en-US/cssref/css_websafe_fonts.html?msclkid=bba5a462b0de11ec818276f93e804f77)

---

basicamente existem 3 nome que vc pode usar que vai deixar generico (não é uma fonte): sans-serif

todos os dispositivos vão ler, ele vai escolher qual fonte sem serifa que ele vai colocar  
outra é serif, ele vai colocar uma fonte serifada

outra é monospace, vai colocar uma fonte monoespaçada

os tipos não foram especificados, então o navegador vai escolher uma fonte que ele considere padrão

aperte Ctrl + espaço em font-family pra poder ver todas as fontes e suas combinações

font-family: Ctrl + espaço;

sempre termine a sua safe combination com um nome generico

nomes genericos: sans-serif, serif, monospace etc.

---

## TAMANHO DE FONTE

pra mudar o tamanho da fonte usamos o font-size: ;

Medidas Absolutas: cm, mm, in (polegada), px, pt, pc

Medidas Relativas: em, ex, rem, vw, vh, %

Para medida de fonte, o recomendado é usar o 'px' e o 'em', não usar o resto

16px = 1em

É tbm recomendado usar o 'em' ao invés do 'px' pois sera melhor de ver o site em um celular, tablet, notebook etc.

---

---

## PESO DA FONTE

Para mudar o peso da fonte a tag necessaria é a font-weight:

font-weight: ; (font-weight) - peso da fonte

Pesos literais: (para algumas fontes)

lighter, normal, bold, bolder

Pesos numericos: (começa em 100 termina em 900)

100, 200, 300, 400, 500, 600, 700, 800, 900

ver se a fonte tem essas opções/variações de peso acima

---

---

## ESTILO DA FONTE

Para mudar o estilo da fonte basta usar a tag font-style (não existe em todas):

font-style: ; (font-style): muda o estilo da fonte

vc pode fazer combinações entre o font-style e o font-weight

---

---

## SUBLINHADO

Para usar o sublinhado use a tag text-decoration:

text-decoration: ; (text-decoration): deixa sublinhado

---

## SHORTHAND FONT

junta todas as configurações de font em uma só:

ordem:

font-style → font-weight → font-size → font-family

---

shorthand font:

font: ; (font): junta todas as características de uma fonte

font: \*font-style → font-weight → font-size → font-family\*;

ex:

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

---

propriedade shorthand: possibilita dizer várias coisas em apenas uma linha

não existe shorthand pra tudo, porém existe vários shorthands que é possível de usar no código HTML

---

## FONTES EXTERNAS

### Google Fonts

Site para fontes:

<https://fonts.google.com/>

obs: NÃO exagere, normalmente é usada apenas 3 ou 4 tipos de fontes, um pra grandes títulos, outro pra destaque, e uma fonte normal

---

---

outro site para fonts:

<https://www.dafont.com/pt/>

se preocupar em ver se a fonte tem acentuações habilitadas

se preocupar em ver se tem minúsculas ou maiúsculas habilitadas

Para usar a fonte que vc baixou no seu html, basta usar a regra (ou seletor especial) `font-face`

`@font-face {}` (`font-face`) : possibilita usar fontes que estejam no pc

```
@font-face {  
  font-family: f;  
  src: url();  
}
```

tem navegadores que funcionam melhor com otf e outros funcionam melhor com ttf

Tipos de `format()`:

opentype (otf)

truetype (ttf)

embedded-opentype

truetype-aat (Apple Advanced Typography)

svg

geralmente o opentype é otf e o truetype é ttf

O `@font-face` "cria" a fonte e não aplica, para aplicar é necessário botar o nome que vc quer pra fonte em `font-family`; e depois botar esse nome no `font-family` do seletor

O recomendado é usar fontes do google-fonts, pra deixar mais leve e padronizado, mas nada impede de misturar as duas ou usar o outro tipo

---

---



## Fontes em imagens

1° lugar

<https://www.whatfontis.com/>

2° lugar

<https://www.fontsquirrel.com/?msclkid=76077811b2b811ec8dc653ab479d3929>

3° lugar

<https://www.myfonts.com/>

---

## Alinhamentos

Para deixar um pouco pra dentro usa o text-indent: ; muito usado em redações

text-indent: ; - deixa um pouco pra dentro

text-indent: 20px; - deixa 20 pixels pra dentro

---

## Seletores Personalizados (Id)

id=" " (id) : identificação

<h1 id=" "> : identificando o h1

<h1 id="aqui pode botar qualquer nome">

<h1 id="principal"> : agora ele esta identificado como "principal"

---

Para personalizar qualquer tag no css que esteja com identificador basta usar o nome da tag + #:

```
h1#principal {  
  
}
```

não é obrigatório usar o nome da tag quando for personalizar no CSS, basta apenas usar o nome do identificador que você deu:

```
#principal {  
}
```

A # identifica id:

em HTML é id = em CSS é # - (só pode um elemento)

O . identifica class:

em HTML é class = em CSS é . (ponto) - (pode múltiplos elementos)

---

```
h1#principal {  
} ← seletor personalizado
```

Dentro de um mesmo documento não pode usar o mesmo id mais de uma vez

---

## Diferenças entre id e class

Se você quiser usar a mesma personalização para mais de uma tag, basta usar o class

id é único

class pode ser múltiplo

---

## class

class=" " (class) :

<h2 class=" "> : classificando os h2s que tenham a mesma característica

<h2 class="aqui pode botar qualquer nome">

<h2 class="basico"> : agora a classe dele é "basico"

\*pode ter vários elementos (tags) com a mesma classe

---

não de nome para suas classes ou seus ids de acordo com a forma e sim a função (função que exerce)

```
h2.basico {  
  
}
```

No class geralmente não botamos o nome do elemento, apenas o . (ponto) e o nome que a gente deu

```
.basico {  
  
}
```

---

Com o class é possível botar quantos tipos de classe quiser:

<h2 class="basico destaque"> : a classe dele é "basico" e "destaque" ou seja, as duas configurações (personalização) serão aplicadas no h2

---

Vc tbm tem a possibilidade de usar um id e um class na mesma tag:

```
<h1 id="principal" class="destaque">
```

ou um id e varios class:

```
<h1 id="principal" class="basico destaque">
```

O id sempre vai ter preferencia, ou seja, a personalização do id sempre vai sobrepor a personalização do class

---

## Organização documento CSS

Primeiro configura o generico (tag sem id nem class):

```
h1 {  
  
}
```

Depois configura a tag que esta usando o id ou o class:

```
h1#principal {  
}
```

O que vale é a última configuração (personalização) ou seja, o que vai ser aplicado é o último a ser ligado

Sempre vai herdar o que veio da camada acima

---

height: ; - altura

width: ; - largura

border: ; - borda

border: 1px solid black; ← ShortHand

↑     ↑     ↑

largura   tipo   cor - da borda

display: inline-block; - tudo na mesma linha

display: none; - não mostra

display: block; - mostra

---

## Pseudo-classes em CSS

# = id

. = class

: = pseudo-classe

---

Pseudo-classes tem que estar relacionadas a um elemento ou a uma classe. Pseudo-classes estão relacionadas ao estado de um determinado elemento, consegue configurar o estado.

---

## Pseudo-Classes:

hover : quando eu passar o mouse por cima

```
div:hover {
```

```
    background-color: yellow;
```

```
} ← Quando eu passar o mouse por cima vai ficar com o fundo amarelo
```

Isso ↑ é uma configuração de estado: ele só vai ficar amarelo quando eu passar o mouse em cima

Aqui ↑ estamos mexendo na propria div

---

div > p : um paragrafo que esta dentro de uma div, ou seja, um paragrafo que é filho de uma div

ele não vai mexer em todos os paragrafos, ele só vai mexer no paragrafo ou paragrafos que estejam dentro de uma div.

---

```
div > p {
```

```
    display: none;
```

```
} ← não mostre nada
```

Os paragrafos que estão dentro da div eu não vou querer mexer (não vão aparecer)

---

```
div:hover > p {
```

```
} ← quando eu passar o mouse em cima, o paragrafo que esta dentro...
```

↓

```
div:hover > p {
```

```
    display: block;
```

```
} ← vai ser exibido na tela ( mostrado )
```

↓

```
div:hover > p {  
    display: block;  
    color: white;  
} ← vai ficar com a cor branca
```

↓

```
div:hover > p {  
    display: block;  
    color: white;  
    background-color: red;  
} ← vai ficar com o fundo vermelho
```

↓

```
div:hover > p {  
    display: block;  
    color: white;  
    background-color: red;  
    width: 170px;  
} ← vai ficar com a largura de 170 pixels
```

Aqui ↑ estamos mexendo em um paragrafo que esta dentro da div

**\*\*Aqui começamos a pensar na criação de menus, é um prototipo inicial no raciocinio que precisamos ter de elementos escondidos e elementos exibidos\*\***

---

---

## Pseudo-elementos

a:visited (visited) : quando for visitado

```
a:visited {  
}
```

```
a:visited {  
color: darkred;  
} ← quando for visitado vai ficar com a cor vermelho escuro
```

---

a:active (active) : quando for clicado

```
a:active {  
}
```

```
a:active {  
color: yellow;  
} ← quando for clicado vai ficar com a cor amarela
```

pseudo-elemento mexe diretamente no conteúdo do elemento, já as pseudo-classes definem estados de elementos dentro da html

---

pseudo-elementos são identificados pelos dois pontos depois da tag

a::after (after) : depois de todos

```
a::after {  
content: ' [link]';  
} ← depois de todos os links vai aparecer a mensagem '[link]'
```

---

a::before (before) : antes de todos

```
a::before {  
content: '→';  
} ← antes de todos os links vai aparecer uma setinha
```

---

## Links

```
a {  
  color: red;  
  text-decoration: none;  
  font-weight: bold;  
}
```

```
a:visited {  
  color: darkred;  
}
```

```
a:hover {  
  text-decoration: underline;  
}
```

---

## seletores personalizados 2.0

# = id

. = class

: = pseudo-class

:: = pseudo-element

> = children

---



# Modelo De Caixas (box model)

Uma caixa ela tem um tamanho, esse tamanho é especificado em duas medidas muito importante e com palavras que se parecem muito

1º Medida: Altura = height

2º Medida: Largura = width

---

Podemos também traçar uma linha em volta da box, ela vai se chamar border

uma border é uma linha que circunda o seu conteúdo



---

Para desfocar essa linha que está grudada com o pontilhado usamos o padding que significa preenchimento.

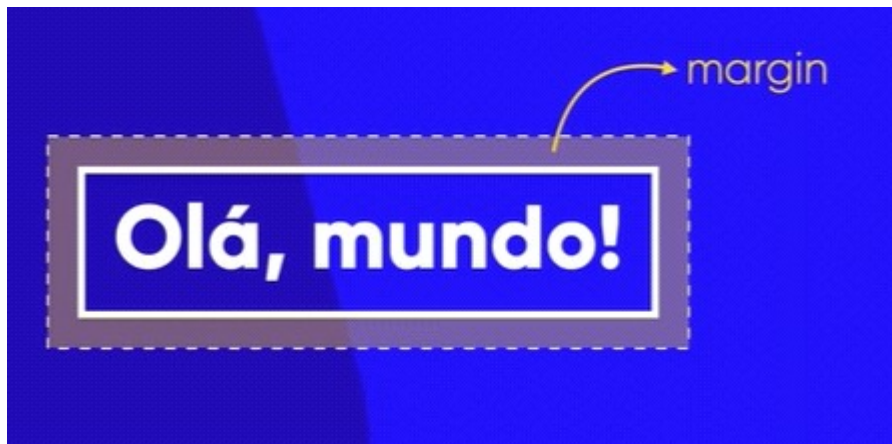
O padding pode ser calculado em qualquer dimensão, pode ter um padding em cima, em baixo, dos lados ou igual

## Modelo de Caixas



---

Podemos criar um espaço externo, esse espaço se chama margin  
da borda pra dentro chamamos de padding e da borda pra fora chamamos de margin



---

outra coisa que podemos fazer dentro da margem, do lado de fora da borda é criar um traçado que se chamara outline

borda é dentro do elemento e o traçado é fora

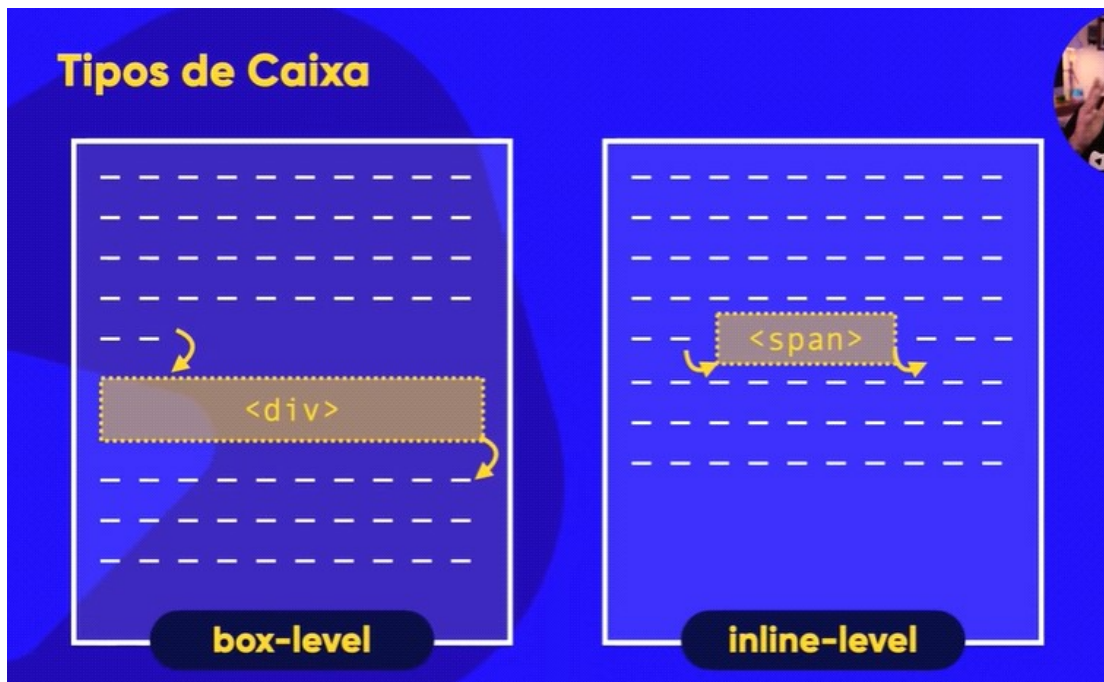


## Tipos De Caixa

Existem dois tipos de caixas, a box-level e a inline-level

um box-level sempre se inicia numa linha nova, e também sempre ocupa a largura total da tela, ou seja, a largura dele é sempre 100% da viewport

um inline-level não pula pra próxima linha, ele vai pro lado e forma a caixa exatamente do tamanho pra conter o conteúdo do elemento e quando termina ele não quebra linha ele continua, ele também não ocupa a largura total (inteira), ou seja, ele ocupa exatamente o tamanho que ele precisa



## Tipos de Caixa

```
<div>  
<h1>-<h6>  
<p>  
<main>  
<header>  
<nav>  
<article>  
<aside>  
<footer>  
<form>  
<video>
```

**box-level**

```
<span>  
<a>  
<code>  
<small>  
<strong>  
<em>  
<sup>-<sub>  
<label>  
<button>  
<input>  
<select>
```

**inline-level**

## Diferença entre div e span

O div é quando eu quero quebrar linha, criar um box que ocupe a largura inteira e depois quebrar a linha de novo

O span é quando eu quero um elemento e um próximo elemento do lado

---

---