

# MAPAS MENTAIS

# HTML E CSS

---



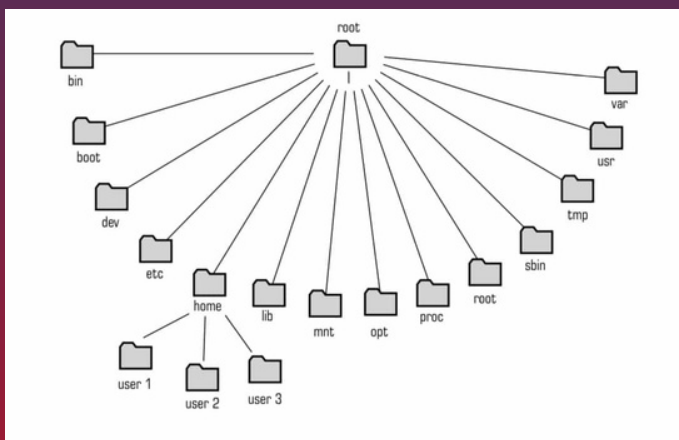
MARIA LEILIANE

# INTRODUÇÃO

Todo **software** é um conjunto de pastas e arquivos, pastas ao nível de organização e arquivos ao nível de execução de comandos e tarefas. Existe uma infinidade de tipos de arquivos, cada um com sua particularidade e objetivo, como as extensões .txt para arquivo de texto, .py para arquivo python, .js para arquivo java script entre outros.

**Diretório** é uma nomenclatura utilizada que significa pasta.

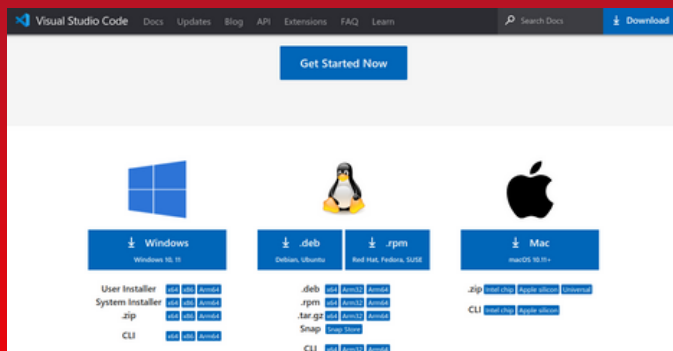
## ESTRUTURA DE PASTAS / DIRETÓRIOS LINUX



**O que é Html e Css?** Tudo referente a web tem html e css, o html insere as informações e o css faz a personalização.

**Significados de html e css,** o html é linguagem de marcação de hipertexto, css folha de estilo em cascata.

É possível editar em qualquer editor de texto, mas existem ferramentas que auxiliam a escrita do código como o studio visual code (**VS code**).



# ESTRUTURA INICIAL DO HTML

**O que é uma tag?** É tudo que esteja envolvido por < (**maior que**) > (**menor que**) o que está escrito no meio é o nome da tag. É possível ter uma tag dentro de outra tag como:

```
<head> <div> Menu </div> </head>
```

Quando a tag tem uma / (**barra**) indica o fechamento da mesma como: </head>

Para criar um documento html é necessário dar um nome e no final colocar a extensão .html ao abrir o documento no VS code (editor de texto) clique em ! (**exclamação**) e em seguida **enter** para o editor entregar a estrutura inicial do html pronta.

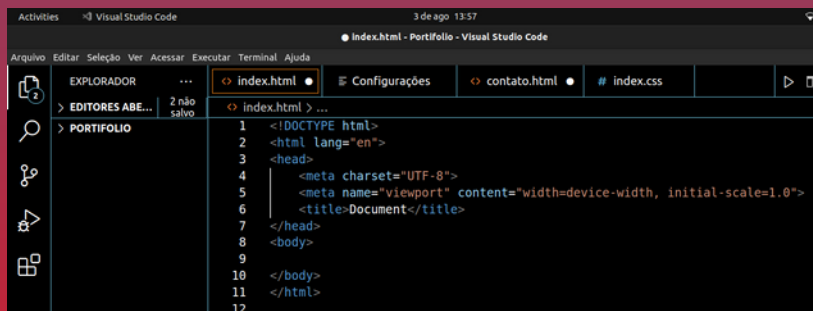
Selecione todas as linhas e aperte a tecla **tab** para fazer a **indentação** e facilitar a leitura das linhas.

**O doctype** não é uma tag é por isso que não possui um fechamento na estrutura vem seguido do **atributo html** que indica qual é a versão de html escrita no documento para que o navegador possa fazer a leitura correta, sem a tag doctype o navegador adota uma leitura padrão de html podendo gerar erros na leitura do código.

**O atributo lang** serve para indicar o idioma como:

pt-br (portugues brasileiro), en (english) entre outros.

**A tag <html>** deve envolver todo documento html, tags não visuais ficam dentro da tag <head> e tags visuais dentro da tag <body>



The screenshot shows the Visual Studio Code interface with a file named 'index.html' open. The editor displays the following HTML code:

```
1 <!DOCTYPE html>
2 <html lang="en">
3 <head>
4   <meta charset="UTF-8">
5   <meta name="viewport" content="width=device-width, initial-scale=1.0">
6   <title>Document</title>
7 </head>
8 <body>
9
10 </body>
11 </html>
12
```

The code is formatted with indentation for the head and body sections. The file explorer on the left shows the project structure with 'index.html' selected.

# TAGS ESTRUTURAIS

---

São **tags visuais** e responsáveis pela **construção inicial** do HTML.

**<head></head>** define o cabeçalho da página e todo conteúdo dentro dela é o cabeçalho.

**<nav></nav>** define o conteúdo de navegação é muito utilizada em conjunto com outras tags, é comum estar dentro da tag head.

**<ul></ul>** define um conjunto de lista não ordenada é comum encontrar dentro da tag nav.

**<ol></ol>** conjunto de lista ordenada onde cada item da lista é levado em consideração, sendo enumerado do primeiro ao último.

**<li></li>** lista de itens é comum ser encontrado dentro da ul definindo cada item da lista.

**<main></main>** dentro da tag fica o conteúdo principal, essa tag é responsável por fazer a página ser encontrada por motores de busca como o **Google**.

**<article></article>** define um artigo da página é usada geralmente em blogs.

**<aside></aside>** colocamos um conteúdo paralelo correlacionado com o conteúdo principal, mas que não é tão importante.

**<footer></footer>** é o rodapé da página pode conter navegação ou não, geralmente tem links uteis como rede sociais, copyright e logo da empresa.

**<div></div>** é uma tag genérica e não possui valor semântico e serve para criar divisões na página.

# ATRIBUTOS

---

Servem principalmente para **personalizar e adicionar funcionalidades** em uma tag. Existem atributos específicos de uma tag e os atributos incomuns como os apresentados a baixo.

Atributo **class** serve para se comunicar com a tag dentro do css ou do js pode ter mais de uma tag com a mesma classe.

Atributo **id** serve para se comunicar com a tag só pode ser usado em uma única tag.

Atributo **style** serve para personalizar dentro do código html escrevendo css dentro do código.

Atributo **lang** indica a linguagem.

Atributo **title** o nome da janela no navegador.

Atributo **alt** titulo alternativo para a tag geralmente usado para descrever imagens funciona para acessibilidade.

Atributo **hidden** indica se a tag vai ficar invisível não é necessário passar nenhum valor.

Atributo **aling** define o padrão de alinhamento.

Atributo **width** define a largura.

Atributo **height** define a altura.

# TITULOS E TEXTOS

Do `<h1> ... <h6>` temos as prioridades de títulos, cada uma com uma estilização padrão sendo `<h1></h1>` para o texto principal.

São muito importantes para os motores de busca.

`<span></span>` é um texto que não é título nem parágrafo, apenas um texto "jogado" não tem estilização padrão.

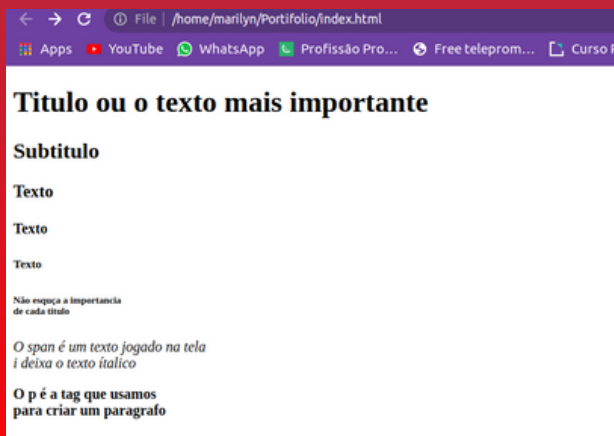
`<i></i>` o texto fica em itálico.

`<b></b>` deixa em bloco (negrito).

`<p></p>` indica um parágrafo.

`<br/>` break row quebra linha uma tag que abre e fecha nela mesma.

```
<? index.html x # index.css •
<? index.html > ...
1  <!DOCTYPE html>
2  <html lang="en">
3  <head>
4      <meta charset="UTF-8">
5      <meta name="viewport" content="width=device-width, initial-scale=1.0">
6      <title>Document</title>
7  </head>
8  <body>
9      <div class="textos">
10         <h1>Titulo ou o texto mais importante</h1>
11         <h2>Subtitulo</h2>
12         <h3>Texto</h3>
13         <h4>Texto</h4>
14         <h5>Texto</h5>
15         <h6>Não esqueça a importancia<br/> de cada titulo</h6>
16
17         <span>
18             <i>
19                 | O span é um texto jogado na tela<br/> i deixa o texto italico
20             </i>
21         </span>
22
23         <p>
24             <b>
25                 | O p é a tag que usamos<br/> para criar um paragrafo
26             </b>
27         </p>
28     </div>
29 </body>
30 </html>
```



# FORMULARIOS

Utilizado para inserir dados em um campo de input.

Campos de input não são necessariamente utilizados somente dentro da tag **<form>** **</form>** dentro da tag tem o **atributo action** é para onde as informações inseridas são enviadas.

A tag **<label>****</label>** é uma espécie de título para o campo input, dentro tem o **atributo for** que serve para quando clicar em cima da label automaticamente o campo de input é selecionado.

Na tag **<input>****</input>** o **atributo id** serve para estilização e para ser identificado dentro do **atributo for** da label. O atributo **type** serve para indicar o tipo do nosso input.

O **atributo name** indica a chave do input. O **atributo placeholder** é um texto auxiliar e assim que o usuário começa a digitar o texto some.

```
<? index.html X
<? index.html > ...
1  <!DOCTYPE html>
2  <html lang="en">
3  <head>
4      <meta charset="UTF-8">
5      <meta name="viewport" content="width=device-width, initial-scale=1.0">
6      <title>Document</title>
7  </head>
8  <body>
9
10 <form action="/index.html">
11 <label for="email">Email</label>
12 <input style="display: block;" id="senha" type="email" name="email"
13 placeholder="Coloque um email valido"/>
14
15 <label for="senha">Senha</label>
16 <input style="display: block;" id="senha" type="password" name="senha"/>
17
18 <label for="text">Texto</label>
19 <input style="display: block;" id="text" type="text" name="text" />
20
21 <label for="checkbox">Checkbox</label>
22 <input style="display: block;" id="checkbox" type="checkbox" name="checkbox" />
23
24 <label for="radio">Masculino</label>
25 <input style="display: block;" id="radio" type="radio" name="genero" value="Masculino" />
26
27 <label for="radio">Feminino</label>
28 <input style="display: block;" id="radio" type="radio" name="genero" value="Feminino" />
29
30 <label for="radio">Outro</label>
31 <input style="display: block;" id="radio" type="radio" name="genero" value="Outro" />
32
33 <label for="range">Range</label>
34 <input style="display: block;" id="radio" type="range" name="range" />
35
36 <label for="file">Arquivo</label>
37 <input style="display: block;" id="file" type="file" name="file" />
38
39 <label for="numero">Numero</label>
40 <input style="display: block;" id="numero" type="number" name="numero" />
41
42 <label for="data">Data</label>
43 <input style="display: block;" id="data" type="date" name="date" />
44
45
46 <select style="display: block;" name="Dia da semana">
47 <option value="Seg">Segunda</option>
48 <option value="Ter">Terça</option>
49 <option value="Qua">Quarta</option>
50 <option value="Qui">Quinta</option>
51 <option value="Sex">Sexta</option>
52 <option value="Sab">Sabado</option>
53 <option value="Dom">Domingo</option>
54 </select>
55
56 <textarea style="display: block;" name="mensagem id mensagem" rows="4"></textarea>
57 <button>Enviar</button>
58 </form>
59
60 </body>
61 </html>
```

# TIPOS DE INPUTS

**email** serve para validar se o conteúdo enviado é um email.

**password** oculta visualmente os dados.

**text** não tem nenhuma característica especial, serve apenas para inserir um texto podendo conter palavras ou números.

**checkbox** é uma caixa de checagem onde você seleciona, muito utilizada em aceitação de termos.

Temos três inputs do **radio** e serve para escolha de um único valor dentre outros a resposta enviada é sempre o value.

**range** é utilizado para o usuário setar um valor de 0 a 100 zero para o início da barra e 100 para a barra arrastada até o final.

**file** é utilizado para que o usuário possa enviar um arquivo.

**number** serve para enviar um número.

**date** serve para que o usuário escolha facilmente uma data porque abre a opção de um calendário.

O **select** não é uma tag de input, mas serve para inserir dados selecionando uma opção.

**textarea** serve para o usuário poder enviar uma mensagem maior.

**button** envia os dados do formulário disparando o atributo action.

The image displays a vertical stack of various HTML input types:

- Email:** A text input field containing the email address "leiliane.lopez@hotmail.com".
- Senha:** A password input field with masked characters "\*\*\*\*\*".
- Texto:** A standard text input field containing the text "SÃO BERNARDO DO CAMPO".
- Checkbox:** A checkbox input field that is checked, with a blue checkmark.
- Masculino:** A radio button input field.
- Feminino:** A radio button input field.
- Outro:** A radio button input field.
- Range:** A range input field represented by a horizontal slider bar.
- Arquivo:** A file input field with a "Choose File" button and the text "No file chosen".
- Numero:** A number input field containing the value "22".
- Data:** A date input field showing "mm/dd/yyyy" and a calendar icon.
- Segunda:** A select dropdown menu with "Segunda" selected.
- Area de texto:** A text area input field containing the text "Para escrever... 1234".
- Enviar:** A submit button.



# TAGS DE MÍDIA

São tags de inserção de **imagem video e audio**

Na tag **img** o **atributo src** indica o caminho da imagem, o **./** abre a pasta e seleciona a imagem, caso ele não esteja dentro da mesma pasta que nosso arquivo use **../** para abrir uma pasta dentro de outra pasta ou **../** para voltar uma pasta antes da pasta atual.

Na tag **video** coloca o **atributo controls**, dentro da tag abre outra com o nome **source** coloca o **atributo src./** selecione o video dentro da pasta da mesma forma que fizemos na imagem indique o atributo **type=mp4** para identificar o tipo de video inserido.

O **atributo controls** serve para garantir que teremos acesso a todos dos controles do video. Caso o navegador não tenha suporte para reproduzir o video, a mensagem aparece.

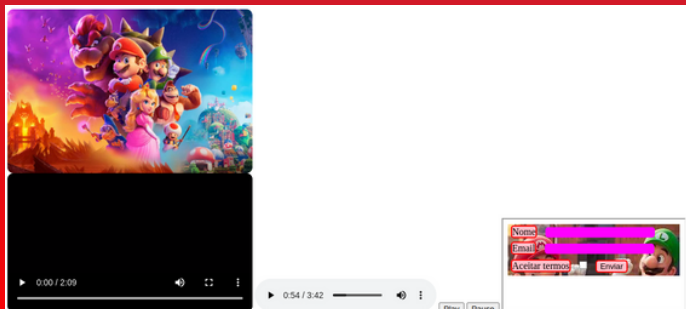
Na tag **audio** coloque o **atributo controls** dentro da tag abre outra com o nome **source** com o **atributo src./** abre a pasta e seleciona o áudio da mesma forma que foi feito na imagem.

O **button** esta disparando uma função dizendo para o documento pegue esse áudio por **id** e de um **play** ou um pause *(isso já é uma funcionalidade js)*.

O **iframe** não é uma tag de mídia, mas serve para reproduzir outra página html dentro da página atual.

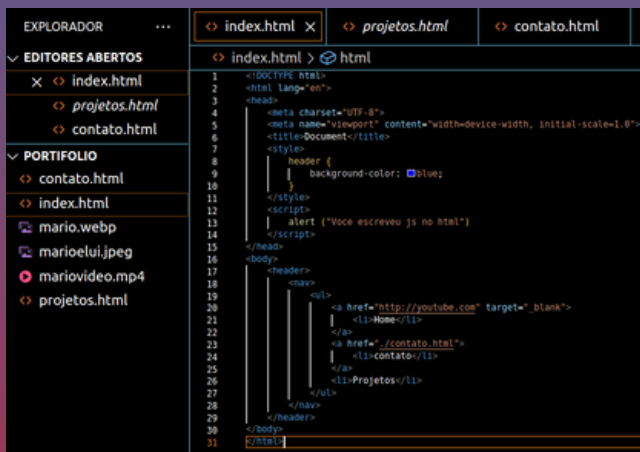
```
<> index.html x <> contato.html ● # index.css

<> index.html > ...
1 <!DOCTYPE html>
2 <html lang="en">
3 <head>
4   <meta charset="UTF-8">
5   <meta name="viewport" content="width=device-width, initial-scale=1.0">
6   <title>Document</title>
7 </head>
8 <body>
9   
10
11   <video controls style="width: 30%; border-radius: 10px;" >
12     <source src="../mariovideo.mp4" type="video/mp4" />
13     O navegador não tem suporte para video!
14   </video>
15
16   <audio id="player" controls>
17     <source src="../audio.mp3" type="audio/mp3"/>
18     O navegador não tem suporte para audio!
19   </audio>
20   <button onclick="document.getElementById('player').play()">Play</button>
21   <button onclick="document.getElementById('player').pause()">Pause</button>
22   <iframe src="../contato.html"></iframe>
23 </body>
24 </html>
```



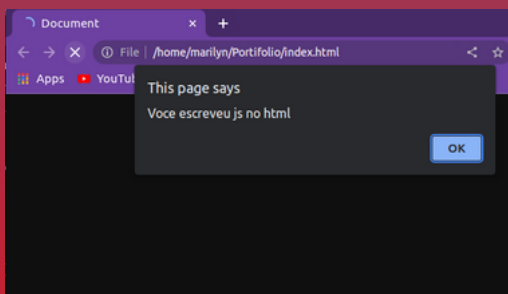
# STYLE, SCRIPT E ÂNCORA

A tag **style** escreve css dentro do html aqui você chama o nome da tag coloca {} e escreve dentro das chaves a **funcionalidade css**



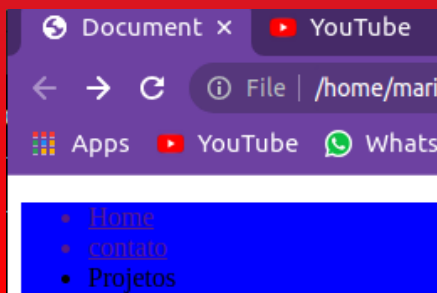
```
1 <!DOCTYPE html>
2 <html lang="en">
3 <head>
4   <meta charset="UTF-8">
5   <meta name="viewport" content="width=device-width, initial-scale=1.0">
6   <title>Document</title>
7   <style>
8     | header {
9       |   background-color: blue;
10    | }
11  </style>
12  <script>
13    | alert ("Voce escreveu js no html")
14  </script>
15 </head>
16 <body>
17   <header>
18     <nav>
19       <ul>
20         <li><a href="http://youtube.com" target=".blank">
21           | <i>Home</i></li>
22         <li><a href="/contato.html">
23           | <i>contato</i></li>
24         <li><a href="/contato.html">
25           | <i>contato</i></li>
26         <li>Projetos</li>
27       </ul>
28     </nav>
29   </header>
30 </body>
31 </html>
```

A tag **script** escreve js dentro do html pode ser colocada no **<heard>** ou **<body>** dentro da tag script colocamos a **funcionalidade alert** de js.



Não é a forma correta de adicionar css ou js dentro do documento html porque **o código fica bagunçado**, o correto linkar o código css ou js dentro do html, vamos aprender mais a frente.

A tag **ancora <a>** cria a navegação entre páginas e o **conteúdo fica clicável** o **atributo href./** direciona a página do documento, pode ser para qualquer página até mesmo uma pagina fora do documento como <http://google.com> se você quiser abrir em uma nova guia basta adicionar o **atributo target** e passar o valor **blank**.



# SELETORES DE CSS

São utilizados para selecionar o elemento no css ou no js para colocar funcionalidades.

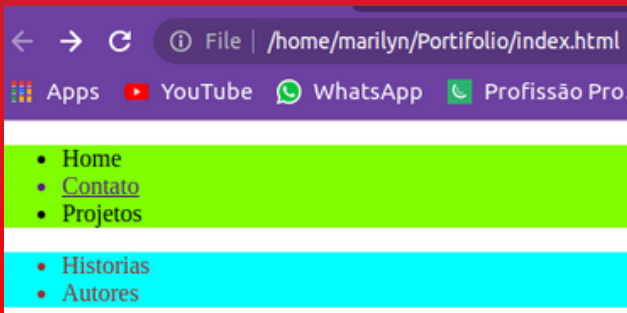
Para selecionar um elemento por nome da tag coloque o **nome da tag** seguido de **{}** e escreva o código dentro das chaves como foi feito no código anterior, não é muito utilizado, pois estiliza todos os elementos que pousem a mesma tag.

Para selecionar pela **class** adicione um ponto e escreva o nome da class, o **ponto é o seletor** de class abre **{}** e escreva as funcionalidades.

É um dos seletores mais utilizados.

O seletor de **id** é **#** abre **{}** e passa as funcionalidades.

```
index.html x projetos.html contato.html
index.html > html
1 <!DOCTYPE html>
2 <html lang="en">
3 <head>
4   <meta charset="UTF-8">
5   <meta name="viewport" content="width=device-width, initial-scale=1.0">
6   <title>Document</title>
7   <style>
8     .header1 {
9       background-color: chartreuse;
10    }
11
12    .header2 {
13      background-color: aqua;
14      color: brown;
15    }
16
17    #seuId {
18      height: 200px;
19    }
20  </style>
21 </head>
22 <body>
23   <header class="header1">
24     <nav>
25       <ul>
26         <li><a href="#">Home</li>
27         <li><a href="/contato.html">Contato</li>
28         <li>Projetos</li>
29       </ul>
30     </nav>
31   </header>
32   <header id="seuId" class="header2">
33     <nav>
34       <ul>
35         <li>Historias</li>
36         <li>Autores</li>
37       </ul>
38     </nav>
39   </header>
40 </body>
41 </html>
```



# LINKS CSS

Para tirar o css do html e linkar o arquivo css dentro do documento html.

Crie um documento com a **extensão .css**

Dentro da **<head>** coloque a tag **<link/>** é uma tag com fechamento nela mesma, dentro da coloque o **atributo rel**, passe o valor **stylesheet**, coloque o **atributo href** acrescente o **/** para chamar o seu documento css.

<> index.html ●	# index.css	<> projetos.html	<> contato.html
-----------------	-------------	------------------	-----------------

```
<> index.html > html
1 <!DOCTYPE html>
2 <html lang="en">
3 <head>
4   <meta charset="UTF-8">
5   <meta name="viewport" content="width=device-width, initial-scale=1.0">
6   <title>Document</title>
7   <link rel="stylesheet" href="index.css" />
8 </head>
9 <body>
10
11 </body>
12 </html>
```

# UNIDADES DE MEDIDAS CSS

Temos dois tipos de unidades de medidas, a **absoluta** e as **relativas**.

A única unidade de medida absoluta é o **píxel** identificado com **px**

O **rem** é uma unidade de **medida relativa** porque depende de outro valor do documento, o tamanho padrão dessa unidade de medida é **16px** sua equivalência é 1,2,3... vezes o valor do html pode utilizar valores fracionados como 2.5

O **em herda** seu tamanho da **tag** que o envolve, sua equivalência é 1,2,3... vezes o valor da tag que envolve.

<> index.html x	# index.css	<> projetos.html
-----------------	-------------	------------------

```
<> index.html > html
1 <!DOCTYPE html>
2 <html lang="en">
3 <head>
4   <meta charset="UTF-8">
5   <meta name="viewport" content="width=device-width, initial-scale=1.0">
6   <title>Document</title>
7   <link rel="stylesheet" href="index.css" />
8 </head>
9 <body>
10   <h1 id="pixel">Aprendendo o pixel</h1>
11   <h1 id="rem">Aprendendo o rem</h1>
12   <div>
13     <h1 id="em">Aprendendo o em</h1>
14   </div>
15 </body>
16 </html>
```

Aprendendo o pixel

Aprendendo o rem

Aprendendo o em

<> index.html	# index.css
---------------	-------------

```
# index.css > #em
1 html {
2   font-size: 20px;
3 }
4
5 #pixel {
6   font-size: 40px;
7   color: blueviolet;
8 }
9
10 #rem {
11   font-size: 2.5rem;
12   color: blue;
13 }
14
15 div {
16   font-size: 15px;
17 }
18
19 #em {
20   font-size: 3em;
21   color: hotpink;
22 }
```

# BOX MODEL

Tudo na construção **web** é feito em **blocos de conteúdos**.

**margin** é o distanciamento de um elemento com relação aos outros elementos que estão fora dele em todas as direções.

**margin-top** distanciamento no topo, **margin-right** distanciamento a direita, **margin-bottom** distanciamento a baixo, **margin-left** distanciamento a esquerda.

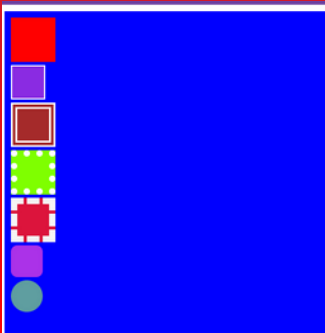
Colocando somente **um valor** no **margin** ele aplica este valor em **todas as direções**. Se colocar somente **dois valores**, o **primeiro** valor faz referência **para cima e para baixo**, o **segundo** valor faz referência **para direita e para esquerda**.

É possível colocar somente o **margin** e escrever os **quatro valores**, por padrão é adicionado as margens na seguinte ordem **topo, direita, baixo e esquerda**.

**padding** é o distanciamento com relação aos **elementos** que estão **dentro**, a aplicação de valores pode ser adicionada da mesma forma que o **margin**.

Tipos de borda **border solid** (solida), **double** (dupla), **dotted** (pontilhado), **dashed** (tracejado) para a borda é necessário discriminar o **tamanho** e a **cor** da borda, ainda dentro da borda temos outra possibilidade o **border-radius** esse é referente ao **arredondamento** da borda passamos o valor do arredondamento em **pixel** para ter um círculo perfeito usamos o **valor de 50%**.

```
1 <!DOCTYPE html>
2 <html lang="en">
3 <head>
4   <meta charset="UTF-8">
5   <meta name="viewport" content="width=device-width, initial-scale=1.0">
6   <title>Document</title>
7   <link rel="stylesheet" href="/index.css">
8 </head>
9 <body>
10   <div class="pai">
11     <div class="filho">
12     </div>
13     <div class="borda">
14     </div>
15     <div class="borda1">
16     </div>
17     <div class="borda2">
18     </div>
19     <div class="borda3">
20     </div>
21     <div class="borda4">
22     </div>
23     <div class="borda5">
24     </div>
25   </div>
26 </body>
27 </html>
```



```
1 {
2   margin: 0;
3   padding: 0;
4   text-decoration: none;
5 }
6 .pai {
7   width: 500px;
8   height: 500px;
9   background-color: #000080;
10  margin: 10px;
11  padding: 0px;
12 }
13 .filho {
14   background-color: #FF0000;
15   width: 50px;
16   height: 50px;
17   margin: 0px;
18   padding: 10px;
19 }
20 .borda {
21   border: 2px solid #FFFFFF;
22   background-color: #800080;
23   width: 50px;
24   height: 50px;
25   margin: 0px;
26 }
27 .borda1 {
28   border: 10px double #FFFFFF;
29   background-color: #8B4513;
30   width: 50px;
31   height: 50px;
32   margin: 0px;
33 }
34 .borda2 {
35   border: 10px dotted #FFFFFF;
36   background-color: #4169E1;
37   width: 50px;
38   height: 50px;
39   margin: 0px;
40 }
41 .borda3 {
42   border: 10px dashed #FFFFFF;
43   background-color: #DC143C;
44   width: 50px;
45   height: 50px;
46   margin: 0px;
47 }
48 .borda4 {
49   border-radius: 10px;
50   background-color: #4682B4;
51   width: 50px;
52   height: 50px;
53   margin: 0px;
54 }
55 .borda5 {
56   border-radius: 50%;
57   background-color: #6495ED;
58   width: 50px;
59   height: 50px;
60   margin: 0px;
61 }
62 }
```

# WIDTH E HEIGHT CSS

**Width** é altura, **height** é a largura de um **determinado elemento**.

Dentro do css temos duas **class**, a primeira **div** é a **class pai** e a segunda **div** é a **class filho**.  
Na class pai temos 60% que corresponde a **60% de largura** do tamanho padrão do documento.

Na class filho temos **50% na altura** e na **largura** que corresponde a **50% da div pai**.

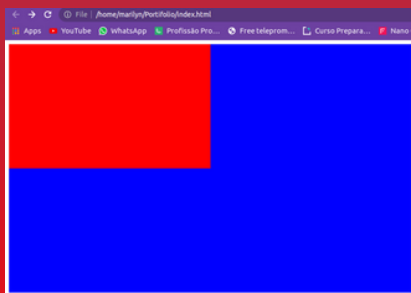
A % é uma unidade de **medida relativa**.

<pre>&lt;&gt; index.html ● # index.css ● &lt;&gt; projetos.htm</pre>	<pre>&lt;&gt; index.html ● # index.css ●</pre>
<pre>&lt;&gt; index.html &gt; html &gt; body &gt; div.pai &gt; di</pre>	<pre># index.css &gt; .filho</pre>
<pre>1 &lt;!DOCTYPE html&gt; 2 &lt;html lang="en"&gt; 3 &lt;head&gt; 4   &lt;meta charset="UTF-8"&gt; 5   &lt;meta name="viewport" content="width=device-width, in 6   &lt;title&gt;Document&lt;/title&gt; 7   &lt;link rel="stylesheet" href="./index.css" /&gt; 8 &lt;/head&gt; 9 &lt;body&gt; 10   &lt;div class="pai"&gt; 11     &lt;div class="filho"&gt; 12   &lt;/div&gt; 13 &lt;/div&gt; 14 &lt;/body&gt; 15 &lt;/html&gt;</pre>	<pre>1 .pai { 2   width: 60%; 3   height: 500px; 4   background-color: blue; 5 } 6 7 .filho { 8   background-color: red; 9   width: 50%; 10  height: 50%; 11 }</pre>

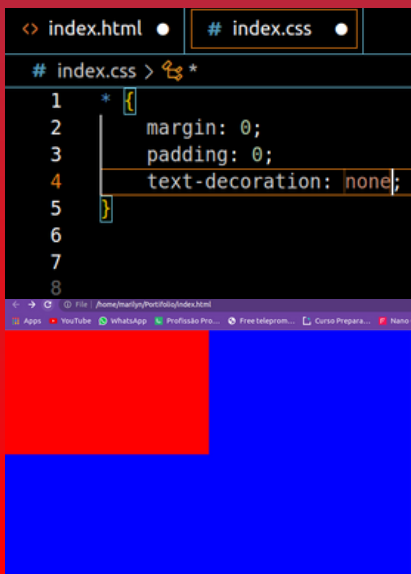
## CSS RESET

Os navegadores entregam **valores padrões de css** para evitar isso é necessário aplicar o **css reset** no começo do documento adicione **\* {}** e escreva o reset dentro das chaves  
Tem **opções prontas** na internet mais complexas, mas com **margin, padding** e **text-decoration** já é uma boa opção de css reset.

### Sem CSS reset



### Com CSS reset



# POSITION

O **position** define como o elemento vai se **posicionar** e se **comportar** em tela, por **padrão** é static que segue o **fluxo normal** dos elementos em tela.

A altura **100vh** significa que o documento **ocupa 100%** dos pixels disponíveis da tela e se adicionar **200vh** duplica o tamanho e adiciona **scroll (rolagem) vertical**, o mesmo serve para **vw** a diferença e que **200vw** adiciona **scroll (rolagem) horizontal**.

**position fixed** (fixo) significa que o **elemento** fica fixo, para **definir a posição** podemos inserir **top** (a cima), **left** (a esquerda), **right** (direita) e **bottom** (a baixo) para definir a quantos pixels de distância o elemento vai ficar.

**sticky** (pegajoso) segue o **fluxo normal** e a partir do momento em que o elemento toca o **topo** do documento ele **gruda e não sai**.

**relative** (relativo) serve para definir um valor em relação à **posição atual** do elemento.

**absolute** pode se comportar de duas maneiras, em um elemento de **position static** ele leva em consideração os **limites da tela**, em um elemento com **position relative** leva em consideração os **limites do elemento**.

```
<? index.html x # index.css
1 index.html > h1 > body >
2 <!DOCTYPE html>
3 <html lang="en">
4 <head>
5   <meta charset="UTF-8">
6   <meta name="viewport" content="width=device-width, initial-scale=1">
7   <title>Document</title>
8   <link rel="stylesheet" href="index.css">
9 </head>
10 <body>
11   <div class="possicao1"></div>
12   <div class="possicao2"></div>
13   <div class="possicao3"></div>
14   <div class="relativi">
15     <div class="possicao4"></div>
16   </div>
17 </body>
18 </html>
```



```
<? index.html # index.css x
# index.css > .possicao4
1 {
2   margin: 0;
3 }
4 body {
5   height: 200vh;
6   width: 200vw;
7 }
8 .possicao {
9   background-color: darkgreen;
10  width: 50px;
11  height: 50px;
12  position: fixed;
13  top: 0px;
14  left: 60px;
15 }
16
17 .possicao1 {
18   background-color: chartreuse;
19   width: 50px;
20   height: 50px;
21   margin-top: 60px;
22   position: sticky;
23   top: 0px;
24 }
25
26 .possicao2 {
27   background-color: deeppink;
28   width: 50px;
29   height: 50px;
30   position: relative;
31   top: 10px;
32 }
33
34 .possicao3 {
35   background-color: indigo;
36   width: 50px;
37   height: 50px;
38   position: absolute;
39   top: 200px;
40 }
41
42 .relativi {
43   background-color: cyan;
44   width: 60px;
45   height: 60px;
46   position: relative;
47   top: 100px;
48 }
49
50 .possicao4 {
51   background-color: blueviolet;
52   width: 40px;
53   height: 40px;
54   position: absolute;
55   left: 10px;
56 }
```

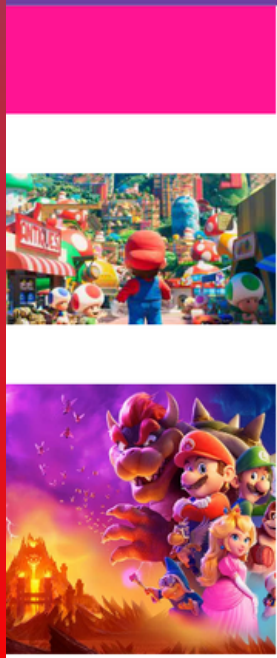
# BACKGROUND

Já conhecemos o **background-color** e somente com a **propriedade background** também é possível adicionar uma cor **ao fundo**.

O **background image** coloca a opção **url()** dentro dos parentes coloca o **link da imagem**, se não quiser usar um link da web é só colocar o **./** e selecionar a imagem **salva** dentro da **pasta**, para a imagem não ficar se **repetindo** adicione o **background-repeat** e seleciona **no-repeat**.

O **background-size** com a opção **contain** deixa a imagem com a melhor **qualidade possível**, a opção **cover** cobre **todo o bloco** não levando em consideração a **qualidade da imagem**.

O **background-position** serve para definir a **posição da imagem** dentro do **bloco**.

<> index.html ●	# index.css	<> index.html	# index.css ✕
<b>&lt;&gt; index.html &gt; ...</b>		<b># index.css &gt; .fundoColorido</b>	
<pre>1 &lt;!DOCTYPE html&gt; 2 &lt;html lang="en"&gt; 3 &lt;head&gt; 4   &lt;meta charset="UTF-8"&gt; 5   &lt;meta name="viewport" content="width=device-width, 6     &lt;title&gt;Document&lt;/title&gt; 7   &lt;link rel="stylesheet" href="./index.css" /&gt; 8 &lt;/head&gt; 9 &lt;body&gt; 10  &lt;div class="fundoColorido"&gt;&lt;/div&gt; 11  &lt;div class="fundoImagem"&gt;&lt;/div&gt; 12  &lt;div class="fundoImagem2"&gt;&lt;/div&gt; 13 &lt;/body&gt; 14 &lt;/html&gt;</pre>		<pre>1 { 2   margin: 0; 3 } 4 body { 5   height: 200vh; 6   width: 200vw; 7 } 8 9 .fundoColorido { 10  background: deeppink; 11  width: 250px; 12  height: 100px; 13 } 14 15 16 17 .fundoImagem { 18   background-image: url(data:image/jpeg;base64); 19   background-repeat: no-repeat; 20   background-size: contain; 21   background-position: 50% 50%; 22   width: 250px; 23   height: 250px; 24 } 25 26 27 .fundoImagem2 { 28   background-image: url(./mario.webp); 29   background-repeat: no-repeat; 30   background-size: cover; 31   width: 250px; 32   height: 250px; 33 } 34</pre>	
			




# CORES

A forma que já conhecemos de chamar uma cor é pelo **nome**, outra forma de fazer isso é usando a forma **hexadecimal** exemplo **#000** cada número representa uma determinada cor, existem ferramentas na internet que ajudam a escolha da cor sem ser necessário decorar todos os códigos ([https://www.w3schools.com/colors/colors\\_picker.asp](https://www.w3schools.com/colors/colors_picker.asp)).

Outra forma de escolher uma cor é através do **rgb** com **três valores** separados por vírgulas para definir a cor.

A **rgba** é uma forma de escolher a cor parecida com a **rgb** a diferença é que usamos **4 valores** e no último parâmetro é possível definir o nível de opacidade da cor.

Pick a Color:



Or Enter a Color:

Or Use HTML5:

Selected Color:

Black Text

Shadow

White Text

Shadow

#cc99ff

rgb(204, 153, 255)

hsl(270, 100%, 80%)

Lighter / Darker:

100%	#ffffff
95%	#f2e6ff
90%	#e6ccff
85%	#d9b3ff
80%	#cc99ff
75%	#bf80ff
70%	#b366ff
65%	#a64dff
60%	#9933ff
55%	#8c1aff
50%	#8000ff
45%	#7300e6
40%	#6600cc
35%	#5900b3
30%	#4d0099

<> index.html x

# index.css

<> index.html > ...

```
1 <!DOCTYPE html>
2 <html lang="en">
3 <head>
4   <meta charset="UTF-8">
5   <meta name="viewport" content="width=device-width, initial-scale=1.0">
6   <title>Document</title>
7   <link rel="stylesheet" href="style.css">
8 </head>
9 <body>
10   <div class="hexadecimal"></div>
11   <div class="rgb"></div>
12   <div class="rgba"></div>
13
14 </body>
15 </html>
```

<> index.html

# index.css x

# index.css > ...

```
1 {
2   margin: 0;
3 }
4 .hexadecimal {
5   background: #000;
6   width: 50px;
7   height: 50px;
8 }
9
10 .rgb {
11   background: rgb(98, 38, 127);
12   width: 50px;
13   height: 50px;
14 }
15
16 .rgba {
17   background-color: rgba(127, 255, 212, 0.671);
18   width: 50px;
19   height: 50px;
20 }
```

# PSEUDO CLASS E PSEUDO ELEMENTO

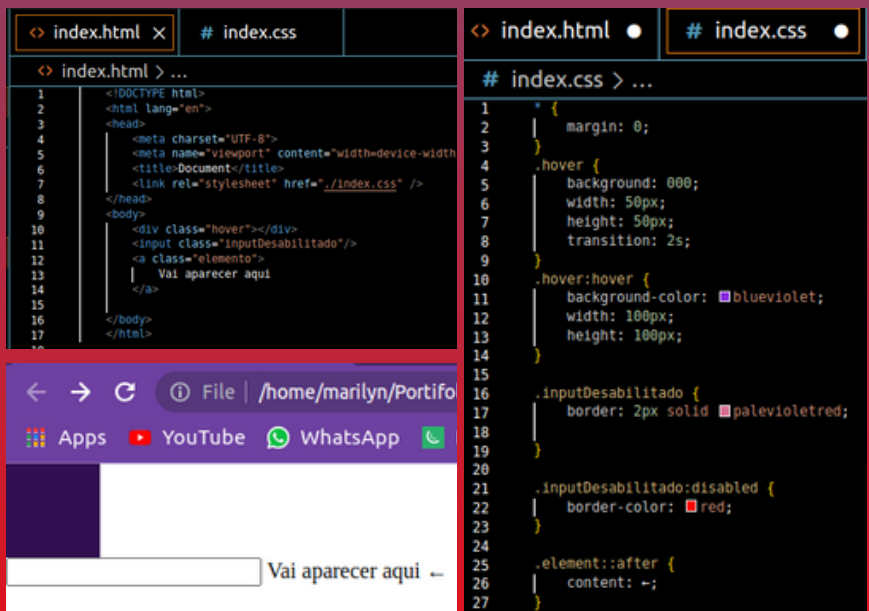
A **pseudo class** é uma palavra é adicionado ao seletor de css para apontar um estado especial.

O **hover** vai acionar uma **funcionalidade especial** quando o mouse for passado por cima, para a transição acontecer de forma sutil deve adicionar um **transition** colocar o **tempo** para acontecer a **transição**.

O **disabled** quando o **input** estiver desabilitado vamos adicionar uma **borda** e quando estiver habilitado a borda **muda de cor**.

A diferença da pseudo class para o pseudo elemento é que a pseudo class **interage** com um **elemento** que **existe**, o pseudo elemento **cria** um **elemento** vamos conhecer o **after** e o **before**.

Na pseudo classe usa um único: no pseudo elemento usa com dois pontos::



# DISPLAY

A opção display serve para determinar como um elemento vai se comportar em tela.


**inline** coloca os elementos lado a lado, somente não é possível atribuir uma altura e uma largura para os elementos.

**inline-block** funciona exatamente como o inline, mas é possível definir uma altura e largura para o elemento.

**block** é como se fosse nosso display padrão e coloca os nossos elementos adiciona uma quebra de linha a cada elemento que colocamos.

**none** é a ausência do display o elemento foi escrito no nosso documento html, mas não aparece em tela.

<> index.html ●	# index.css
<pre>&lt;&gt; index.html &gt; ... 1      &lt;!DOCTYPE html&gt; 2      &lt;html lang="en"&gt; 3      &lt;head&gt; 4          &lt;meta charset="UTF-8"&gt; 5          &lt;meta name="viewport" content="width=device-width, initial-scale=1.0"&gt; 6          &lt;title&gt;Document&lt;/title&gt; 7          &lt;link rel="stylesheet" href="index.css"&gt; 8      &lt;/head&gt; 9      &lt;body&gt; 10         &lt;div class="display1"&gt;01&lt;/div&gt; 11         &lt;div class="display1"&gt;01&lt;/div&gt; 12         &lt;div class="display2"&gt;&lt;/div&gt; 13         &lt;div class="display2"&gt;&lt;/div&gt; 14         &lt;div class="display3"&gt;&lt;/div&gt; 15         &lt;div class="display3"&gt;&lt;/div&gt; 16         &lt;div class="display4"&gt;&lt;/div&gt; 17     &lt;/body&gt; 18 &lt;/html&gt;</pre>	<pre># index.css &gt; ... 1  * { 2      margin: 0; 3    } 4  .display1 { 5      width: 50px; 6      height: 50px; 7      background: #aquamarine; 8      display: inline; 9    } 10 11 .display2 { 12     width: 50px; 13     height: 50px; 14     background: #brown; 15     display: inline-block; 16   } 17 18 .display3 { 19     width: 50px; 20     height: 50px; 21     background: #green; 22     display: block; 23   } 24 25 .display4 { 26     width: 50px; 27     height: 50px; 28     background: #salmon; 29     display: none; 30   }</pre>



# DISPLAY FLEX

O display flex é um pouco mais complexo, sempre vai ser adicionado no bloco pai e vai refletir em todos os blocos filhos.

Uma das propriedades que usamos com display flex é a **flex-direction** por padrão é row (linha) podemos mudar para **row-reverse** e inverter as ordens dos nossos itens.

Outra opção é a **column** (coluna) e ela coloca os itens em coluna é possível aplicar a propriedade **column-reverse**.

O **align-items** alinha os elementos verticalmente.

**justify-content** alinha os itens horizontalmente.

**flex-start** adota o posicionamento inicial.

**flex-end** todos os itens vão para o final.

**center** coloca os itens no centro.

**space-between** separa os elementos ao máximo.

**space-around** aplica um espaçamento nas laterais.

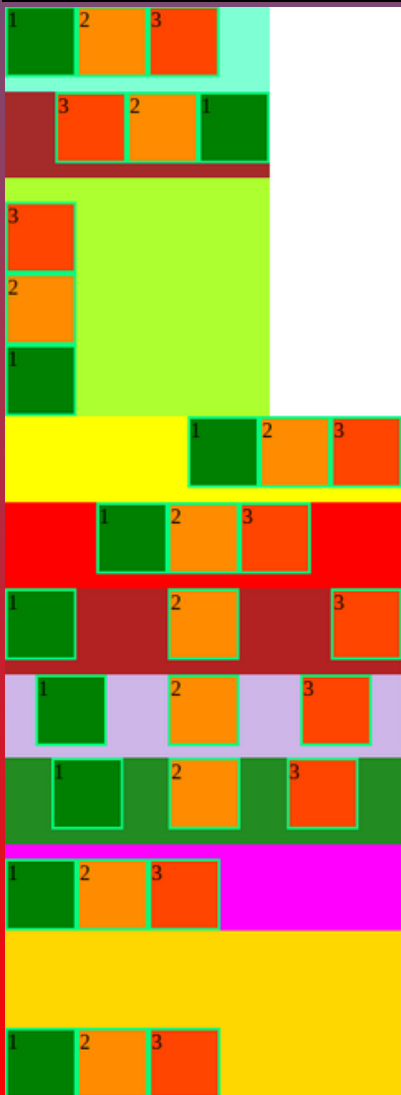
**space-evenly** aplica espaçamento igual em todos os itens e nas laterais.

```
<> index.html x # index.css
<> index.html > html > body > div.pai10
1 <!DOCTYPE html>
2 <html lang="en">
3 <head>
4 <meta charset="UTF-8">
5 <meta name="viewport" content="width=device-width, initial
6 <title>Document</title>
7 <link rel="stylesheet" href="/index.css" />
8 </head>
9 <body>
10 <div class="pai1">
11 <div class="filho1">1</div>
12 <div class="filho2">2</div>
13 <div class="filho3">3</div>
14 </div>
15 <div class="pai2">
16 <div class="filho1">1</div>
17 <div class="filho2">2</div>
18 <div class="filho3">3</div>
19 </div>
20 <div class="pai3">
21 <div class="filho1">1</div>
22 <div class="filho2">2</div>
23 <div class="filho3">3</div>
24 </div>
25 <div class="pai4">
26 <div class="filho1">1</div>
27 <div class="filho2">2</div>
28 <div class="filho3">3</div>
29 </div>
30 <div class="pai5">
31 <div class="filho1">1</div>
32 <div class="filho2">2</div>
33 <div class="filho3">3</div>
34 </div>
35 <div class="pai6">
36 <div class="filho1">1</div>
37 <div class="filho2">2</div>
38 <div class="filho3">3</div>
39 </div>
40 <div class="pai7">
41 <div class="filho1">1</div>
42 <div class="filho2">2</div>
43 <div class="filho3">3</div>
44 </div>
45 <div class="pai8">
46 <div class="filho1">1</div>
47 <div class="filho2">2</div>
```

```
<> index.html # index.css x
# index.css > .pai10
1 {
2   margin: 0;
3 }
4 .pai1 {
5   width: 200px;
6   height: 65px;
7   background: #aquamarine;
8   display: flex;
9 }
10
11
12 .pai2 {
13   width: 200px;
14   height: 65px;
15   background: #brown;
16   display: flex;
17   flex-direction: row-reverse;
18 }
19
20
21 .pai3 {
22   width: 200px;
23   height: 180px;
24   background: #greenyellow;
25   display: flex;
26   flex-direction: column-reverse
27 }
28
29 .pai4 {
30   width: 300px;
31   height: 65px;
32   background: #yellow;
33   display: flex;
34   justify-content: flex-end
35 }
36
37 .pai5 {
38   width: 300px;
39   height: 65px;
40   background: #red;
41   display: flex;
42   justify-content: center;
43 }
44
45 .pai6 {
46   width: 300px;
47   height: 65px;
```

# DISPLAY FLEX

```
48 <div class="filho3">3</div>
49 </div>
50 <div class="pai9">
51 <div class="filho1">1</div>
52 <div class="filho2">2</div>
53 <div class="filho3">3</div>
54 </div>
55 <div class="pai10">
56 <div class="filho1">1</div>
57 <div class="filho2">2</div>
58 <div class="filho3">3</div>
59 </div>
60
61 </body>
62 </html>
```



```
48 background: #firebrick;
49 display: flex;
50 justify-content: space-between;
51 }
52
53 .pai7 {
54 width: 300px;
55 height: 65px;
56 background: #rgb(206, 182, 232);
57 display: flex;
58 justify-content: space-around;
59 }
60
61
62 .pai8 {
63 width: 300px;
64 height: 65px;
65 background: #forestgreen;
66 display: flex;
67 justify-content: space-evenly;
68 }
69
70
71 .pai9 {
72 width: 300px;
73 height: 65px;
74 background: #fuchsia;
75 display: flex;
76 align-items: end;
77 }
78
79
80 .pai10 {
81 width: 300px;
82 height: 200px;
83 background: #gold;
84 display: flex;
85 align-items: center;
86 }
87
88
89 .filho1 {
90 width: 50px;
91 height: 50px;
92 background: #green;
93 border: 2px solid #springgreen;
94 }
95
96 .filho2 {
97 width: 50px;
98 height: 50px;
99 background: #darkorange;
100 border: 2px solid #springgreen;
101 }
102
103 .filho3 {
104 width: 50px;
105 height: 50px;
106 background: #orangered;
107 border: 2px solid #springgreen;
108 }
109 }
```

# DISPLAY GRID

Assim como display flex é inserido no bloco pai, alterando a disposição em tela dos elementos filhos.

**tanplatete-colun** divide os itens em coluna, as colunas podem ser descriminadas em porcentagem ou em píxel o mais comum é em fração, não é necessário criar vários fr é possível colocar **repeat** (3, 1fr) e o documento entende que é dividido em 3 frações.

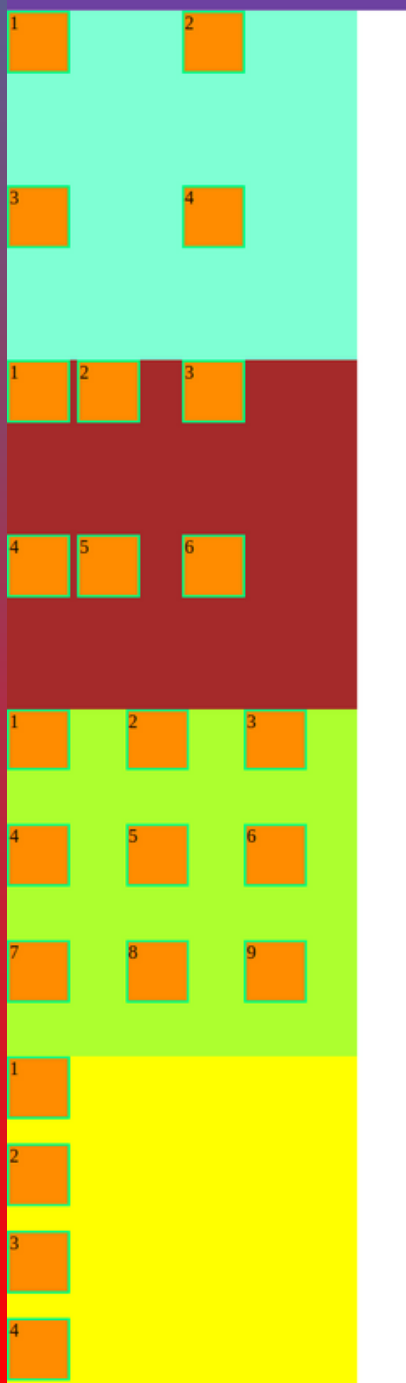
**colun-gap** adiciona um espaçamento entre as colunas.

**display-grid-row** divide os itens em lilhas.

**row-gap** adiciona um espaçamento entre as linhas.

<> index.html X	# index.css ●	<> index.html	# index.css ●
<b>&lt;&gt; index.html &gt; html &gt; body &gt;</b>		<b># index.css &gt; ...</b>	
1	<!DOCTYPE html>	1	{
2	<html lang="en">	2	margin: 0;
3	<head>	3	4
4	<meta charset="UTF-8">	5	.pai1 {
5	<meta name="viewport" content="width=device-width, initial-scale=1">	6	width: 300px;
6	<title>Document</title>	7	height: 300px;
7	<link rel="stylesheet" href="index.css">	8	background: #aquamarine;
8	</head>	9	display: grid;
9	<body>	10	grid-template-columns: 1fr 1fr;
10	<div class="pai1">	11	}
11	<div class="filho">1</div>	12	13
12	<div class="filho">2</div>	14	.pai2 {
13	<div class="filho">3</div>	15	width: 300px;
14	<div class="filho">4</div>	16	height: 300px;
15	</div>	17	background: #brown;
16	<div class="pai2">	18	display: grid;
17	<div class="filho">1</div>	19	grid-template-columns: 20% 30% 50%;
18	<div class="filho">2</div>	20	}
19	<div class="filho">3</div>	21	22
20	<div class="filho">4</div>	23	.pai3 {
21	<div class="filho">5</div>	24	width: 300px;
22	<div class="filho">6</div>	25	height: 300px;
23	</div>	26	background: #greenyellow;
24	<div class="pai3">	27	display: grid;
25	<div class="filho">1</div>	28	grid-template-columns: repeat(3, 1fr);
26	<div class="filho">2</div>	29	column-gap: 5px;
27	<div class="filho">3</div>	30	}
28	<div class="filho">4</div>	31	32
29	<div class="filho">5</div>	33	.pai4 {
30	<div class="filho">6</div>	34	width: 300px;
31	<div class="filho">7</div>	35	height: 300px;
32	<div class="filho">8</div>	36	background: #yellow;
33	<div class="filho">9</div>	37	display: grid;
34	</div>	38	grid-template-rows: repeat(4, 1fr);
35	<div class="pai4">	39	}
36	<div class="filho">1</div>	40	41
37	<div class="filho">2</div>	42	.filho {
38	<div class="filho">3</div>	43	width: 50px;
39	<div class="filho">4</div>	44	height: 50px;
40	</div>		background: #darkorange;
41	</body>		border: 2px solid #springgreen;
42	</html>		}
43			
44			
45			

# DISPLAY GRID



# KEYFREMES

É como fazemos animações com CSS.

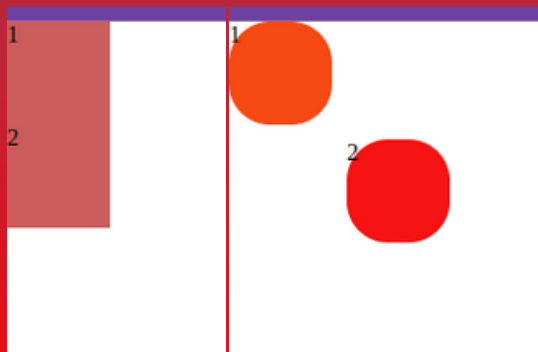
Para chamar o keyframe adiciona um @ e chama animação coloca o nome {} **abre o bloco** e coloca as **funcionalidades**, o **from** vai iniciar com um quadrado, o **to** vai **transformar** em um **círculo**, dentro do bloco da **div** adicione a **funcionalidade animation** e coloque o nome da animação que definida no keyframe e adicione **animation-duration** com a duração de **tempo** para acontecer a animação.

Podemos substituir o **from** e **to** da animação por **porcentagens** e em cada **percentual** acontece uma **mudança diferente**.

Pode ser usada de diversas formas, pode mudar o posicionamento do elemento cor, etc. com essa base já é possível criar animações mais complexas.

```
<> index.html ● # index.css

<> index.html > ...
1      <!DOCTYPE html>
2      <html lang="en">
3      <head>
4          <meta charset="UTF-8">
5          <meta name="viewport" content="wi
6          <title>Document</title>
7          <link rel="stylesheet" href="./in
8      </head>
9      <body>
10         <div class="animal">1</div>
11         <div class="anima2">2</div>
12     </body>
13 </html>
```



```
<> index.html ● # index.css X

# index.css > ...
1  * {
2      margin: 0;
3  }
4  .animal {
5      width: 70px;
6      height: 70px;
7      background: indianred;
8      animation: animacao;
9      animation-duration: 5s;
10 }
11
12 @keyframes animacao {
13     from {
14         border-radius: 0%;
15     }
16     to {
17         border-radius: 50%;
18         background: orangered;
19     }
20 }
21
22 .anima2 {
23     width: 70px;
24     height: 70px;
25     background: indianred;
26     animation: animacao2;
27     animation-duration: 5s;
28     position: absolute;
29 }
30
31 @keyframes animacao2 {
32     0% {
33         top: 0;
34     }
35     10% {
36         top: 20px;
37     }
38     30% {
39         top: 30px;
40     }
41     40% {
42         top: 50px;
43         left: 50px;
44     }
45     50% {
46         top: 80px;
47         left: 80px;
48     }
49     100% {
50         top: 100px;
51         left: 100px;
52         border-radius: 50%;
53         background: red;
54     }
55 }
```



# MEDIA QUERY

**Mobile first** hoje 90% dos sites são acessados por dispositivos móveis, é uma **boa prática** começar o desenvolvimento web pensando nesses acessos.

**Responsividade** é a adaptação que acontece dentro do css para que a pagina seja reproduzida em diferentes tipos de telas.

**Break points** são os pontos de quebra e em cada um desses pontos quando atingimos certa dimensão o site se adapta.

Principais breaks poits (celular) **480px**, (tablet) **768px** e (tela de computador) **1280px**.

Para aplicar no css adicione **@media** coloque **{}** o bloco de código e descreva como o css vai se comportar quando o tamanho for atingido e todo o css dentro do código vai **sobrescrever** o css anterior.

Dentro do media o **min-width** significa que o site começa levando em consideração telas menores e o **max-width** significa que levará em consideração telas maiores.

Para conseguir visualizar a diferença aplicada a cada tipo de tela aperte **f12** e **araste** a separação a esquerda.

<> index.html X	# index.css 3
<> index.html > ...	
1	<!DOCTYPE html>
2	<html lang="en">
3	<head>
4	<meta charset="UTF-8">
5	<meta name="viewport" content="width=device-width, initial-scale=1.0">
6	<title>Document</title>
7	<link rel="stylesheet" href="./index.css" />
8	</head>
9	<body>
10	<div class="responsiv">
11	<div class="bloco">Usuario 1</div>
12	<div class="bloco">Usuario 2</div>
13	<div class="bloco">Usuario 3</div>
14	<div class="bloco">Usuario 4</div>
15	<div class="bloco">Usuario 5</div>
16	<div class="bloco">Usuario 6</div>
17	</div>
18	</body>
19	</html>

# MEDIA QUERY

<> index.html

# index.css X

# index.css > ...

```
1  * {
2  |   margin: 0;
3  | }
4  .responsiv {
5  |   width: 100%;
6  |   height: 1000px;
7  |   background: indianred;
8  |   display: grid;
9  |   grid-template-columns: repeat(1fr);
10 |   column-gap: 10px;
11 |   row-gap: 10px;
12 | }
13
14 .bloco {
15 |   width: 100%;
16 |   height: 100px;
17 |   background-color: aquamarine;
18 | }
19
20 @media(min-width: 400px) {
21 |   .responsiv{
22 |     |   grid-template-columns: repeat(2, 1fr);
23 |     |   background-color: red;
24 |   }
25 | }
26
27 @media(min-width: 768px){
28 |   .responsiv {
29 |     |   grid-template-columns: repeat(3, 1fr);
30 |     |   background-color: yellow;
31 |   }
32 | }
33
34 @media(min-width: 1280px){
35 |   .responsiv {
36 |     |   grid-template-columns: repeat(6, 1fr);
37 |     |   background-color: indigo;
38 |   }
39 | }
40
```

# MEDIA QUERY

