**CSS INTERMEDIÁRIO (IMPORTANTE)**

➤ **DISPLAY:**

➜ **Display Block:** São os displays que usam toda uma linha, não aceitando nada “originalmente” fora ele mesmo na linha. O próximo conteúdo vai p/ a próxima linha.

EX: Os parágrafos <p></p>; as DIVS <div></div>;



.<body>

**<div>**

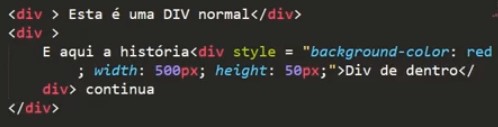
**<div>Esta é uma DIV normal.</div>**

**<div>E aqui a história <div>DIV de dentro</div> continua</div>**

**</div>**

</body>

Aqui, você consegue modificar a largura/altura do elemento, como por exemplo: <div style=“background-color: red; width: 500px;”>





**OBS:** Não é bom ter uma DIV dentro de um parágrafo ou ter um parágrafo dentro de outro parágrafo.

➜ **Display Inline:** São os displays que se obrigam a coexistir na mesma linha, mesmo em uma linha de display block, ele se fará existir nela.

EX: Os spans <span></span>



.<body>

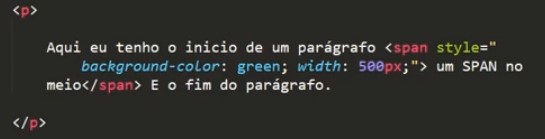
**<p>**

**Aqui eu tenho o início de um parágrafo <span>um SPAN no meio</span> E o fim do parágrafo.**

**</p>**

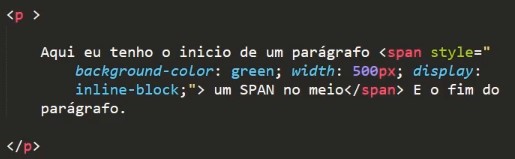
</body>

Já nesse caso, não podemos modificar a largura/altura, mesmo se você colocar o estilo, vai mudar apenas a cor, mas não a altura/largura, como por exemplo: <span style=”background-color: green; width: 500px;”>





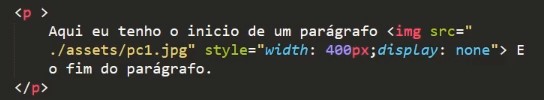
➜ **Display Inline-Block:** Ele vai juntar a característica do INLINE em poder ficar na mesma linha, junto com a característica do BLOCK de poder controlar a LARGURA e a ALTURA.





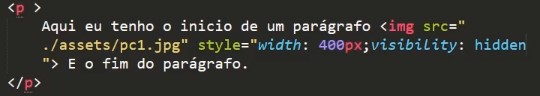
➜ **Display None:** Faz com que o elemento nunca tivesse sido colocado na tela, como se ele nunca tivesse existido lá, pra começo de conversa.







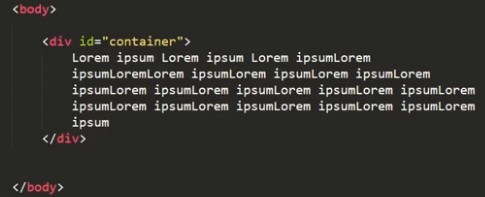
➜ **Visibility:** Como o anterior, ele some com a imagem, mas deixa o espaço que ela ocupava lá! Como se estivesse faltando algo.

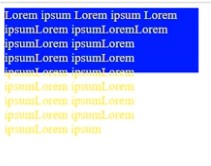




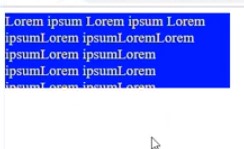
➤ **OVERFLOW:**

➜ **Overflow Visible:** O texto continua apesar do limite do BG que você estabeleceu.

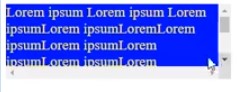




➜ **Overflow Hidden:** O resto do texto é ESCONDIDO, para não ultrapassar o limite do BG.

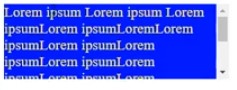


➜ **Overflow Scroll:** Fornece um scroll para você poder rolar o texto.



**OBS:** Apesar de não ser necessário, ele colocou scroll embaixo também.

➜ **Overflow Auto:** Coloca scroll apenas onde for NECESSÁRIO!

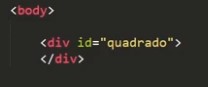


Ou então colocar apenas: overflow-y: scroll;

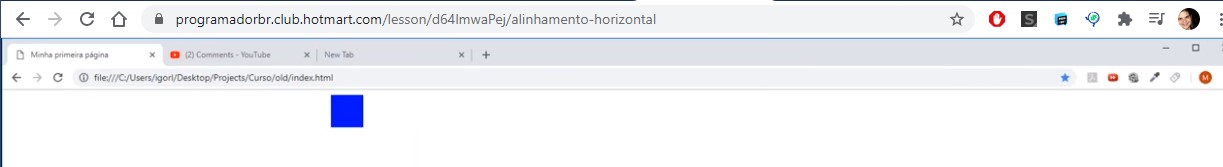
**OBS:** Você pode misturar duas opções também:



➤ **Alinhamento Horizontal:** Escolhemos onde posicionar o elemento (DIV).

Apesar da aparência de centralizado, quando a janela está pequena, quando você abre toda a janela, descobre que o margin-left, nunca deixará centralizado quando você aumenta ou diminui a janela.



Para mudar isso, basta trocar o margin-left, para: margin: auto; dessa forma, seu elemento sempre estará centralizado horizontalmente na tela.

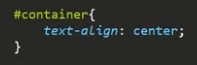
**OBS:** margin-left: auto; → Deixará o quadrado na extrema direita.



**OBS 2:** Para IMAGENS, você precisa primeiramente MUDAR O DISPLAY, sem o display ela nunca mudará de lugar apenas com o margin: auto.

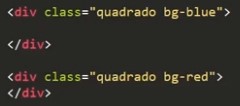
 

**OBS 3:** Mas e se eu não puder mudar o display da imagem? Se eu PRECISAR que ela seja inline-block ou inline? Nesse caso precisarei colocar a imagem dentro de uma DIV:

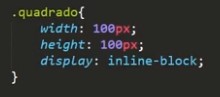
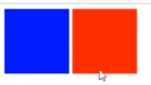




➤ **Position:** É a posição dos elementos.

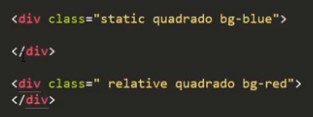
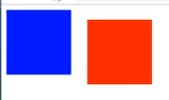
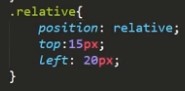
E se eu quiser um do lado do outro? Utilizamos o DISPLAY!

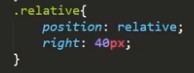
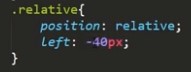
Dessa forma, ele está como position STATIC que é o padrão, mesmo você não tendo escolhido, ele é o position STATIC.

➜ **Position Static:** NÃO MEXE! Ele é estático.

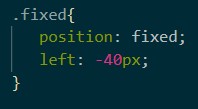
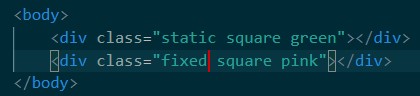
➜ **Position Relative:** A posição relativa, ela é relativa à posição original que ela ficaria caso fosse estática, ou seja, inicialmente ela será igual ao static, mas ela vai poder receber OUTRAS propriedades como: Right/Left/Top/Bottom e aí ele vai dar um distanciamento de acordo com o valor dessas propriedades.

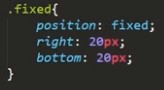
**OBS:** Você também pode escolher ter um quadrado dentro do outro, aplicando:

 **OU**   ➤ 

➜ **Position Fixed:** Ele também poderá receber distanciamentos com Top/Bottom/Left/Right, mas em vez de ser em relação ao elemento caso ele fosse STATIC, será em relação à JANELA do navegador.

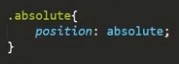
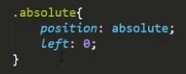
 

Mais exemplos:

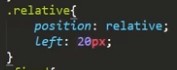
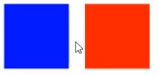
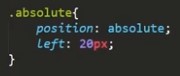
  Vai ficar no canto inferior direito.

**OBS:** No fixo, por mais que você role a janela, o quadrado sempre ficará no canto inferior direito, pois ele é em relação à janela do navegador, independentemente do rolar da página.

➜ **Position Absolute:** É parecido com o STATIC, também podendo receber: Top/Bottom/Left/Right. A diferença é que ele se baseia no PAI do elemento, e se não tiver pai, ele se baseiará no <body>.

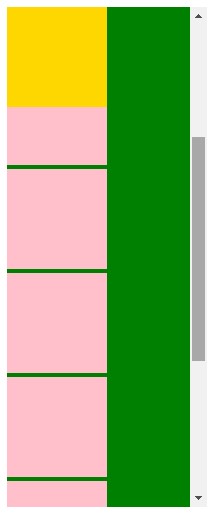
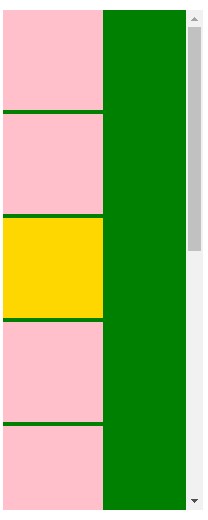
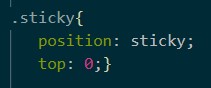
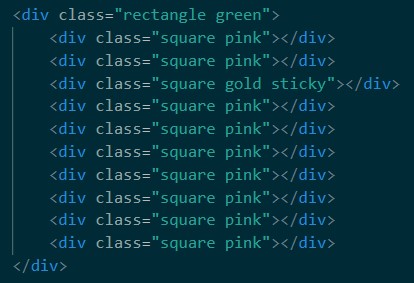
  **OU  **

**OBS:** Então qual a diferença do Relativo para o Absoluto?

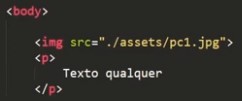
Rel:  Abs:  

**OBS 2:** O Absolute aceita como “Elemento Pai” qualquer elemento, menos o STATIC, se o elemento pai for STATIC, ele vai adotar o pai do pai, se todos forem static, adota o <body>.

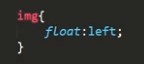
➜ **Position Sticky:** Deixa o elemento sem sair da página, quando você escolhe um lugar para ele, assim que ele tocar esse lugar, ele “gruda” e não sai., mesmo você rolando, e descendo a barrinha, o amarelo vai ficar grudado lá no “top: 0;” que você denominou.



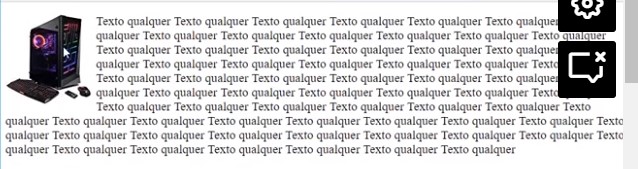
➤ **Float:** É uma propriedade do display que permite que um elemento fique flutuando ao lado de outro elemento.

 SEM STYLE.

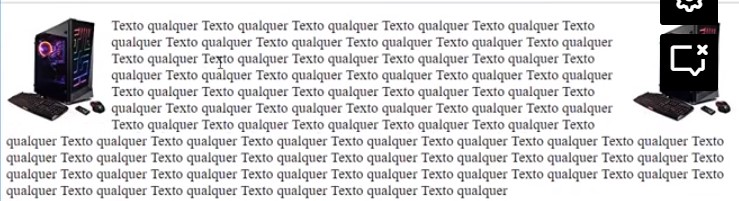
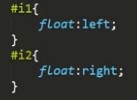
 COLOCANDO DISPLAY no parágrafo.

COLOCANDO FLOAT na imagem.

**OBS:** Se o texto for grande, o texto respeita a imagem, e quando ela termina, ocupa o espaço dela também.

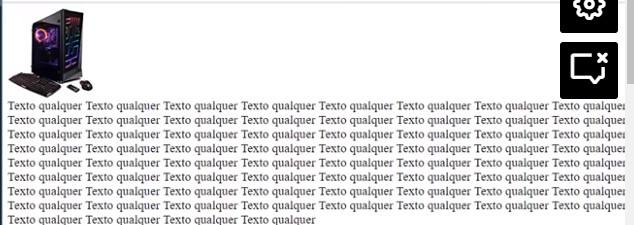
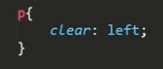


**OU**

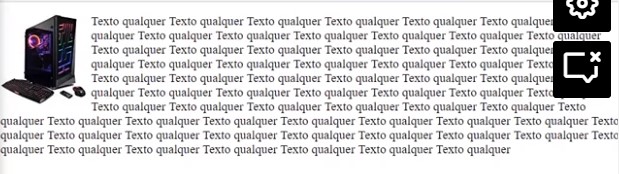
****

➜ **Clear:** Se você usar o clear no texto, ele vai sobrepor o float da imagem e ela não mais ficará ao lado do texto, voltando ao original, como se nunca tivesse tido FLOAT.

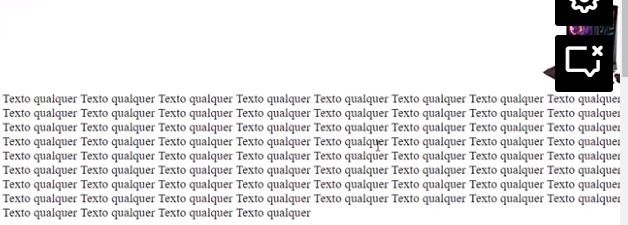
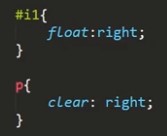
⧫**TIPOS DE CLEAR:** Right **OU** Left **OU** Both **OU** None.



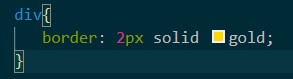
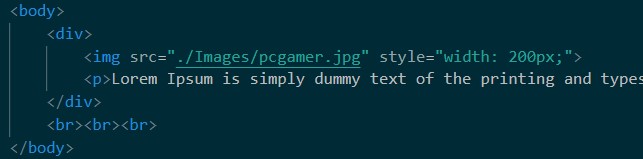
O clear right nesse exemplo não teria diferença, pois não tem elemento flutuando na direita.

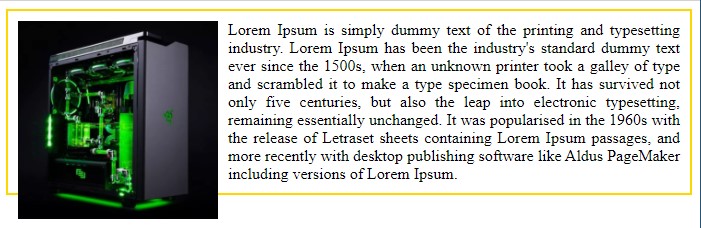


Com elemento flutuando na direita:

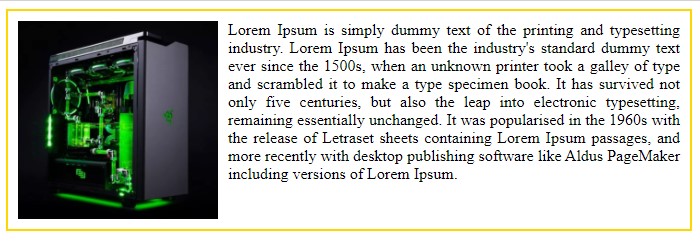
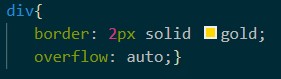


➜ **COLOCANDO DIV:** Você fizer uma div, com os elementos dentro, e colocar uma borda, a imagem vai ultrapassar a borda:

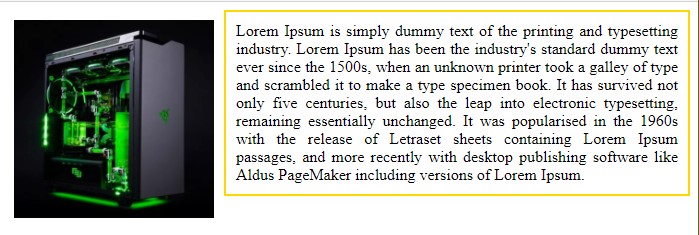
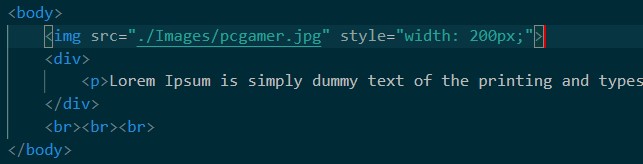




⧫ Mas conseguimos arrumar isso colocando Overflow na DIV:



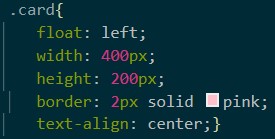
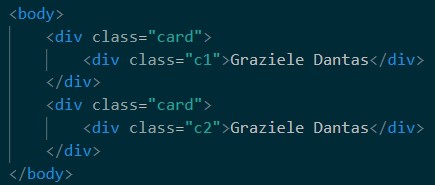
⧫ A imagem também poderia estar fora da div:

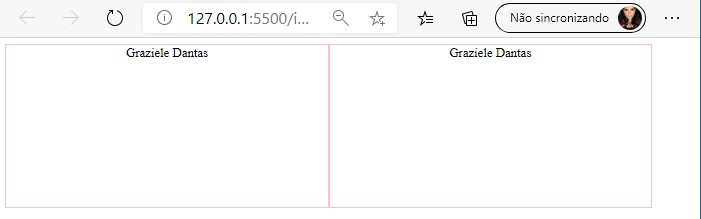


➜ **MENUS:** Também cria-se menus com FLOAT:



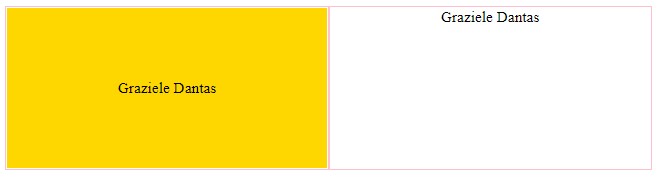
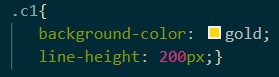
➤ **Alinhamento Vertical:**



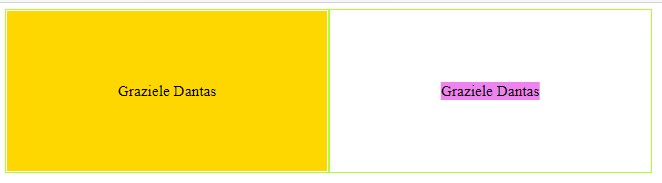
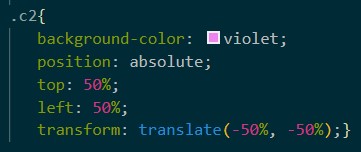
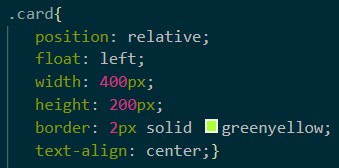


⧫ E como deixar o nome centralizado no meio verticalmente?

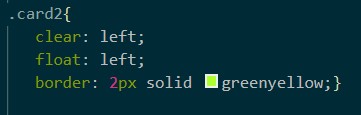
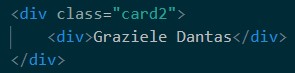
➜ **1º FORMA:** Nessa primeira forma, você precisa saber qual o height desse cartão, para ele funcionar perfeitamente no centro, o que é o caso nesse exemplo, sabemos o height.

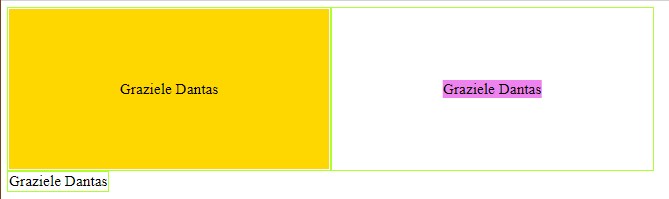


➜ **2º FORMA:** Nessa forma, você precisa mudar o primeiro card para: POSITION RELATIVE, depois fazer todas as mudanças no segundo card, descritas abaixo:



➜ **3º FORMA:** Usando PADDING:

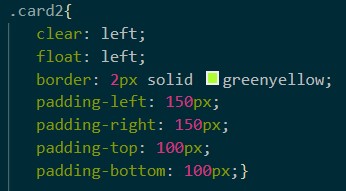




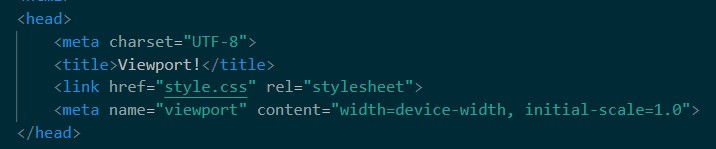
⧫**CLEAR LEFT** - Faz o terceiro card ir para baixo dos dois cards, em vez de ficar do nado direito.

⧫**FLOAT LEFT -** Faz a borda ficar do tamanho do nome, em vez de ocupar a página toda.

USANDO O PADDING, você consegue deixar o 3º cartão parecido com os outros 2:

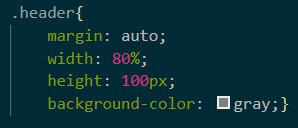
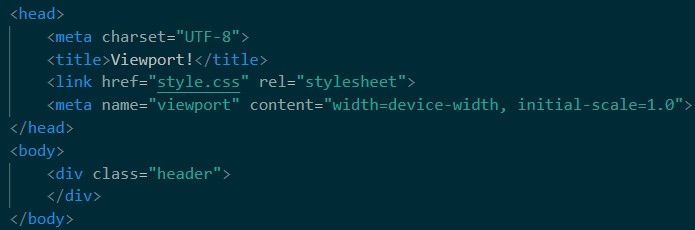


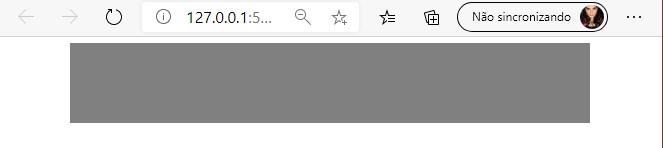
➤ **Viewport:** Cada celular tem um tamanho diferente, então se você programar para um modelo, pode ser que no outro modelo não fique bom. Para arrumar isso:

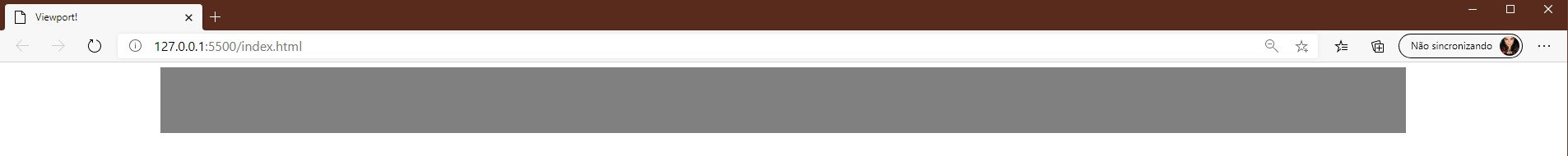


<meta name="viewport" content="width=device-width, initial-scale=1.0">

➤ **Responsividade:** São as medidas de ajuste que você faz para AJUSTAR na página, de acordo com o tamanho da janela, se você aumenta ou diminui, a proporção continua a mesma.

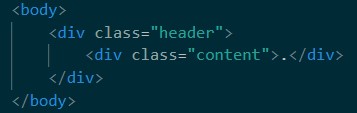






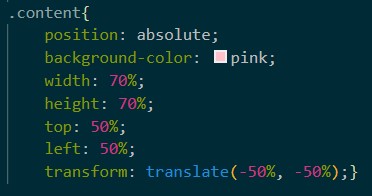
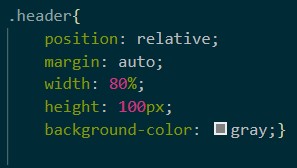
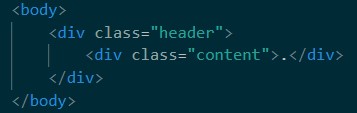
⧫Você também poderia tornar responsivas, uma faixa dentro da outra:

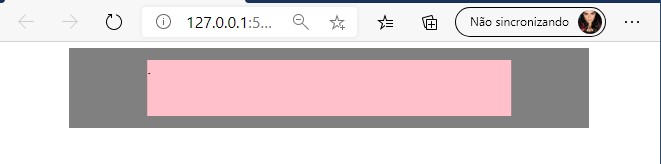
➜ **1º FORMA:** Com Padding.





➜ **1º FORMA:** Caso você precise de medidas EXATAS:

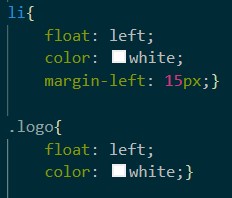
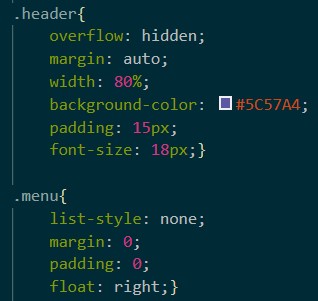
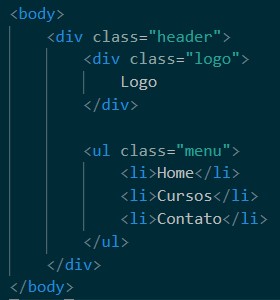


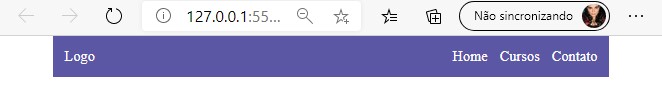


**OBS: FONT-SIZE VW**: É um tipo de fonte com responsividade, ou seja, a fonte aumenta/diminui de acordo com a janela do navegador quando aumentada/diminuída.



➤ **Media Queries:** Novamente em relação à diferença do acesso no browser ou no celular, você pode fazer uma “preparação” para quando o acesso for em telas menores de 700px, 500px, etc.





⧫**RELEMBRANDO**:

➜ **.header:**

⧫Overflow-hidden; → Vai deixar a barra mais larga para os 2 elementos flutuantes.

⧫Margin: auto; → Vai deixar centralizado.

⧫Padding: 15px; → Ajuda na largura da barra.

➜ **.menu:**

⧫List-style: none; → Tira as bolinhas da lista.

⧫Margin/Padding: 0; → As listas tem um padrão que em caso de menu é bom tirar.

⧫Float: right; → Vai flutuar do lado direito.

➜ **li:**

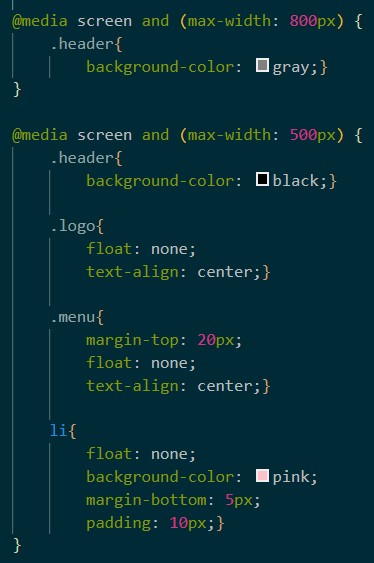
⧫Float: left; → Vai flutuar do lado esquerdo. OBS: Se não tivesse o float teria que ter display: inline-block, mas o float já dá essa característica, então não precisa.

➜ **.logo:**

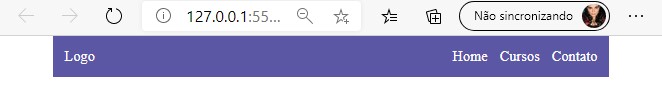
⧫Float: left; → Vai flutuar do lado esquerdo. OBS: Se não tivesse o float teria que ter display: inline-block, mas o float já dá essa característica, então não precisa.

**Media Queries:** Aqui, você vai abrir um painel dentro do css, onde você vai estipular valores para telas com o máximo de 400px por exemplo, onde vai poder dar todo um estilo diferente para essas telas menores, se você for estilizar vários tamanhos, sempre colocar na ordem do maior pro menor (QUANDO MAX-WIDTH): 800px, 500px, 300px, etc.

Se utilizar MIN-WIDTH, é o contrário.



⧫NORMAL, SEM MEDIA QUERIE:



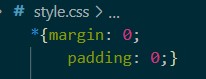
⧫MEDIA QUERIE DE 500 até 800px (MÁX 800px):



⧫MEDIA QUERIE DE MÁX 500px:



**OBS: \***: Colocar \* no style.css, faz todos os elementos da página obedecer:

: NENHUM ELEMENTO terá margin e padding.