

CONTINUAÇÃO HTML & CSS LAYOUT



DEVinHouse

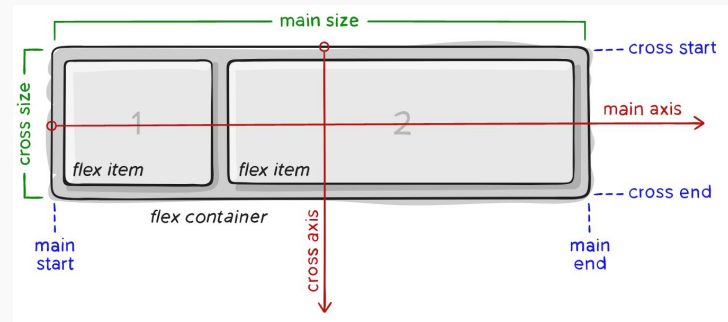
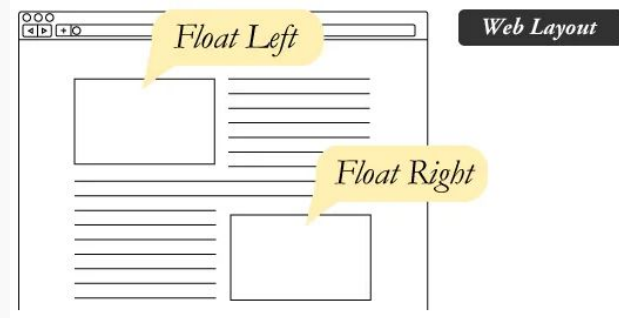
Parcerias para desenvolver a sua carreira

SENAI

<LAB365>

AGENDA | M1S01 - Aula 4

- Revisão e mais elementos HTML
- Revisão e aprofundamento CSS
 - Box model
 - Seletores (*Selectors*)
 - Layout (*render flow*)
 - Flutuar (*float*)
 - Posicionamento (*Position*)
 - Flexbox (*display: flex*)
 - Seletor mídia (*media query*)



- Inspeccionando elementos HTML e estilos CSS
- Alterando propriedades HTML E CSS

HTML



ELEMENTOS DE “CONTAINER”

- **<h1>** Título grande **</h1>** ... ao ... **<h6>** Título menor **</h6>**
- **<div>** Conteúdo em bloco **</div>**
- **<section>** Conteúdo em bloco **</section>**
- **** Conteúdo em linha ****
- **<p>** Parágrafo **</p>**
- **** Texto em *itálico* ****
- **** Texto em negrito ****
- **<hr/>** ⇒ separador **
** ⇒ quebra de linha (“break”)
- etc...
- **** Listas sem ordenação ****
- **** Listas ordenadas ****
- **** Elemento de lista ****

ELEMENTOS DE FORMULÁRIO (INPUT)

- `<input type="text">`
- `<input type="password">`
- `<input type="email">`
- `<input type="number">`
- `<input type="date">`
- `<input type="radio">`
- `<input type="range">`
- `<input type="checkbox">`
- `<input type="color">`

The image displays a variety of HTML input elements arranged in a grid-like fashion. The top row features three text input fields labeled 'Text', 'Password' (with a masked view), and 'Email Address'. The second row shows a 'Number' input, a 'Search' input, and a 'URL Address' input. The third row includes a date input with a calendar icon, a date input with a text mask 'mm/dd/yyyy', and a week input with a text mask 'Week --, ----'. Below these are three interactive elements: a 'Radio Button' with a selected radio button, a 'Range' input with a slider, and a 'Checkbox' with an unchecked checkbox. The bottom row contains three buttons: a 'Submit' button, a 'Reset' button, and a 'Color' input field represented by a black square. The entire collection is titled 'HTML Input Types' in large, bold, pink letters at the bottom of the image.

HTML Input Types

OUTROS ELEMENTOS DE FORM / TABELA

- `<label></label>`
- `<textarea></textarea>`
- `<select></select>`
- `<button></button>`
- `<form></form>`

- `<table>` Tabela `</table>`
- `<tr>` Linha da tabela ("row") `</tr>`
- `<th>` Coluna de cabeçalho ("header") `</th>`
- `<td>` Coluna de dados ("data") `</td>`

```
<table>
  <tr>
    <th>Nome</th>
    <th>Sobrenome</th>
  </tr>
  <tr>
    <td>Rachel</td>
    <td>Green</td>
  </tr>
  <tr>
    <td>Ross</td>
    <td>Geller</td>
  </tr>
</table>
```

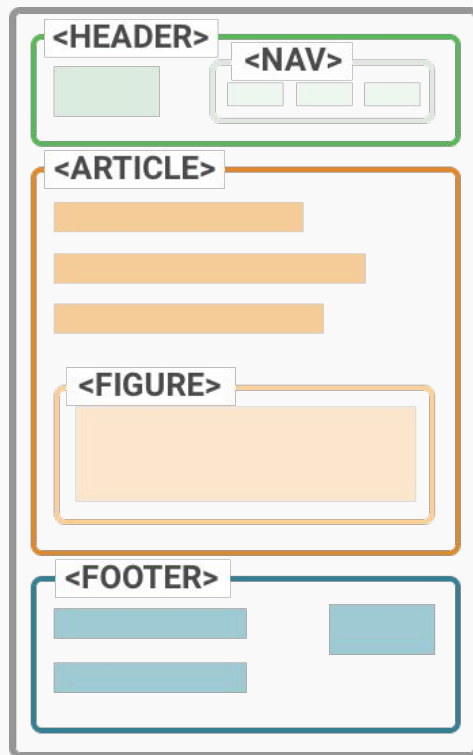
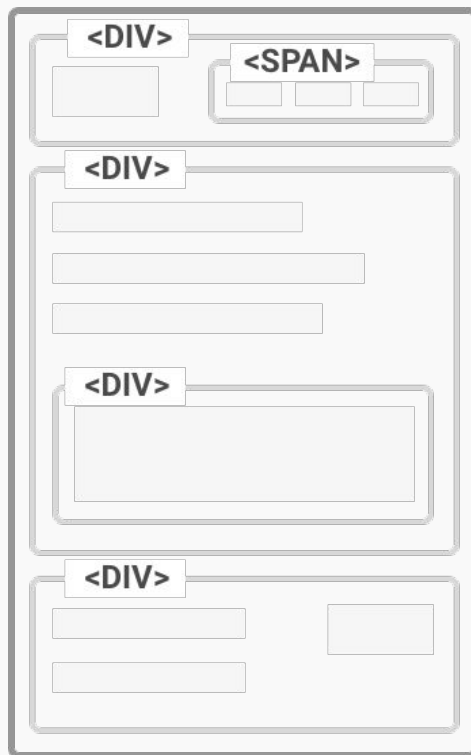
TAGS SEMÂNTICAS DO HTML 5

A tag **<div>** é a forma mais genérica de envolver um conteúdo.

Conforme os sites ficaram mais robustos novas formas de lidar com os problemas surgem.

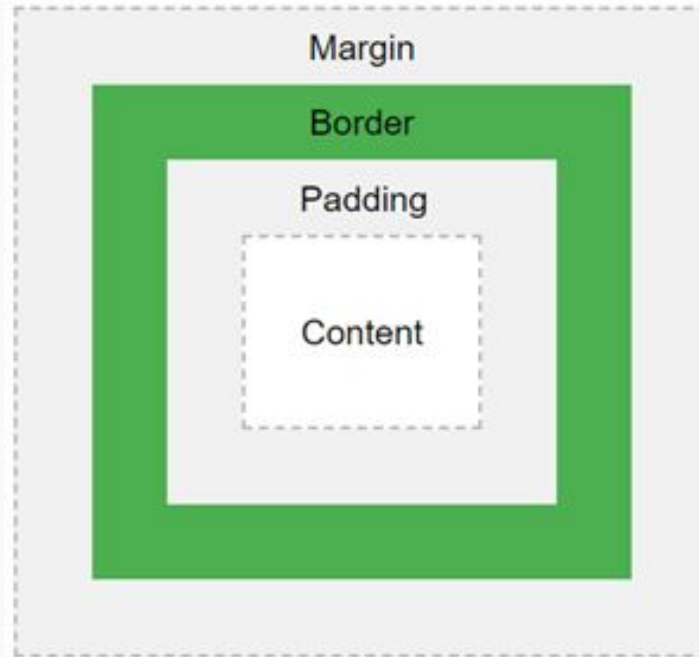
As tags semânticas tem o papel de tornar a escrita de HTML mais **significativa**, ou seja menos genérica.

Existem muitos elementos semânticos, os principais são:
<header> **<main>** **<footer>** **<nav>**
<aside> **<article>** **<figure>**





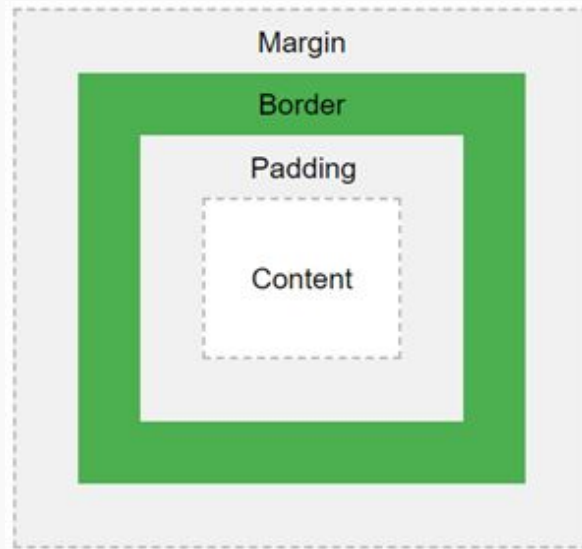
CSS BOX MODEL



CSS BOX MODEL

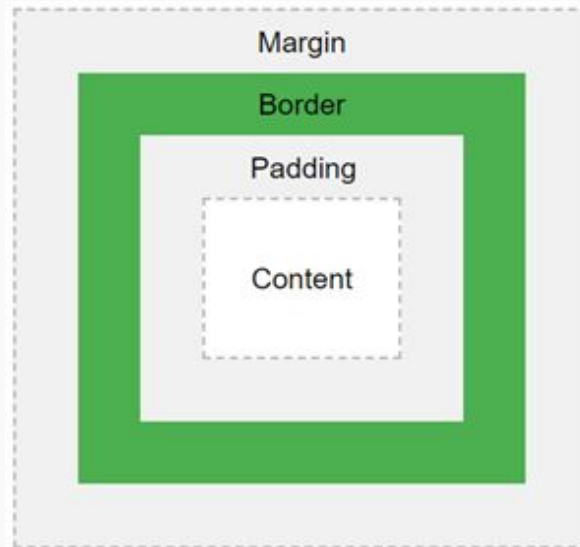
Cada elemento HTML é uma caixa.

- **Content (Conteúdo):** Conteúdo da "caixa", ou seja: textos, imagens.
- **Padding (Preenchimento):** Espaço em torno do conteúdo, transparente.
- **Border (Borda):** Circunda o *padding*.
- **Margin (Margem):** Espaço em volta da borda, transparente.



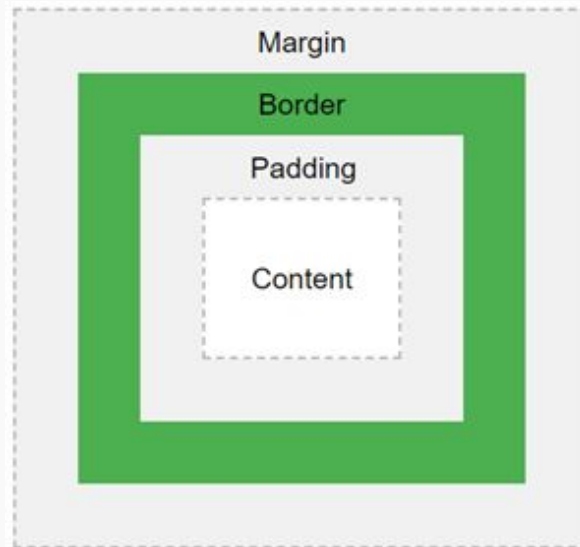
CSS BOX MODEL

- **Content:** largura x altura
 - width: **50px**;
 - height: **80px**;
- **Padding:** *top right bottom left*
 - padding: **5px**;
 - padding: **10px 5px 10px 5px**;
 - padding: **10px 5px**;
 - padding-bottom: **10px**;



CSS BOX MODEL

- **Border:** espessura estilo cor
 - `border: 1px solid black;`
 - `border: 2px dashed red;`
 - `border-top: 80px;`
- **Margin:** *top right bottom left*
 - `margin: 5px;`
 - `margin: 10px 5px 10px 5px;`
 - `margin-left: 10px;`



SELETORES CSS BÁSICOS

- `div` seletor por **TagName** (nome da tag)
Apenas texto com nome do elemento
- `#titulo` seletor por **Id** (atributo id)
Texto com uma **#** (cerquilha, "hash") como prefixo
- `.ativo` seletor por **ClassName** (nome da classe)
Texto com um **.** (ponto) como prefixo
- `*` seletor universal
Asterisco, seleciona todos os elementos

SELETORES CSS BÁSICOS | EXEMPLOS

```
div { ... }
```

Seletor para **TagName** <div>
Ex: <div>...</div>

```
#titulo { ... }
```

Seletor para **Id** "titulo"
Ex: <h1 id="titulo">...</h1>

```
.ativo { ... }
```

Seletor para **ClassName** "ativo"
Ex: <h1 class="ativo">...</h1>

```
div.ativo { ... }
```

Seletor para **TagName** <div> e com **ClassName** "ativo"

```
#titulo.ativo { ... }
```

Seletor para **Id** "titulo" e com **ClassName** "ativo"

```
div#titulo.ativo { ... }
```

Seletor para **TagName** <div> que tenha com **Id** "titulo" e **ClassName** "ativo"
Ex: <h1 id="titulo" class="ativo">...</h1>

```
* { ... }
```

Seletor para **todos** os elementos

INTERVALO DE AULA

DEV!

Finalizamos o nosso primeiro período de hoje. Que tal descansar um pouco?!

Nos vemos em 20 minutos.

Início: 20:20

Retorno: 20:40



SELETORES CSS POR ATRIBUTOS

- `[href]` seletor por **attribute** (atributo HTML) seleciona elementos que possuam o atributo *href*
- `[href="https://google.com"]` seletor por atributo igual elementos com *href* exatamente igual a "https://google.com"
- `[href*="google"]` seletor por atributo parcialmente elementos com *href* que contenha "google" em algum lugar
- `[class~="google"]` seletor por atributo (palavras, " ") elementos com *href* que contenha a palavra "google"

SELETORES CSS POR ATRIBUTOS

- `[href$=".com"]` seletor por atributo com final específico seleciona elementos em que *href* termine com ".com" ou seja igual
- `[href^="http"]` seletor por atributo com início específico elem. com *href* idêntico a "http" ou que comece com "http"
- `[href|="http"]` seletor por atributo com início específico elem. com *href* idêntico a "http" ou que comece com "http-" (hífen)
- `[attr="value" i]` case-**insensitive** (o "i" ou "I" ao final) não se importa se as letras são maiúsculas ou minúsculas
`[attr="value" s]` case-**sensitive** (o "s" ou "S" ao final) considera tudo até mesmo se as letras são maiúsculas ou minúsculas

SELETORES CSS POR ATRIBUTOS | EXEMPLOS

```
img[src*="casa"] { ... }
```

Seletor para **TagName** com um **Attribute** "src" que contenha "casa" em alguma parte de seu valor

Ex:

```
img.reta[src*="casa"] { ... }
```

Seletor para **TagName** de **ClassName** "reta" com um **Attribute** "src" que contenha "casa" em alguma parte

Ex:

```
#titulo[src*="CasA" s] { ... }
```

Seletor para **Id** "reta" com um **Attribute** "src" que contenha "CasA" em alguma parte (case-sensitive)

Ex:

SELETORES CSS COMBINADORES

- `ul li` (" ") seletor de descendentes (filhos, netos...) seleciona todos elementos `li` que sejam descendentes de `ul`
- `ul>li` (">") seletor de filhos (children) seleciona todos elementos `li` que sejam filhos diretos de `ul`
- `ul~li` ("~") seletor geral de irmãos seleciona todos elementos `li` precedidos por um irmão `ul`
- `ul+li` ("+") seletor de irmãos imediatos seleciona apenas elementos `li` imediatamente após um irmão `ul`

SELETORES CSS COMBINADORES | EXEMPLOS

```
div p { ... }
```

Seletor para **TagName** <p> dentro de **TagName** <div>
Ex: <div><section><p>...</p></section></div>

```
div > p { ... }
```

Seletor para **TagName** <p> filho de **TagName** <div>
Ex: <div><p>...</p></div>

```
div ~ p { ... }
```

Seletor de <p> irmão posterior a <div>
Ex: <div></div><section></section><p>...</p>

```
div + p { ... }
```

Seletor de <p> irmão imediato de uma <div>
Ex: <div></div><p>...</p><section></section>

```
div > section + p { ... }
```

Seletor de <p> irmão imediato de uma <section> que seja filha imediata de uma <div>
Ex: <div><section></section><p>...</p></div>

SELETORES CSS PSEUDO-CLASSES

- `div:hover` quando o mouse está em cima de um elemento seleciona todos elementos **div** em que o mouse esteja sobre
- `input:focus` para quando o elemento está focado seleciona o elementos **input** que esteja em foco (selecionado)
- `p:first-child` seletor de primeiros filhos seleciona elementos **p** primeiros filhos de outro elemento
- `p:last-child` seletor de últimos seleciona elementos **p** último filhos de algum elemento

SELETORES CSS PSEUDO-CLASSES

- `p:nth-child(6)` seletor de filho através de regra seleciona elementos **p** que sejam 6º filhos de algum elemento
- `p:only-child` seletor de filhos únicos seleciona elementos **p** que sejam filhos únicos de outro elemento
- `p:not(.inativo)` para excluir, criar exceções seleciona elementos **p** que não tenham classe "inativo"
- `a:visited` seletor links que já foram visitados seleciona elementos **a** cujo link já foi visitado pelo navegador

SELETORES CSS PSEUDO-ELEMENTS

- `p::before` elemento fictício ao início do elemento selecionado cria/estiliza um elemento fictício início do conteúdo de **p**
- `p::after` elemento fictício ao final do elemento selecionado cria/estiliza um elemento fictício ao final do conteúdo de **p**
- `p::first-line` estiliza apenas a primeira linha de um elemento seleciona a primeira linha de **p**
- `a::first-letter` estiliza apenas a primeira letra de um elemento seleciona a primeira letra de **p**

Além de estilos, CSS também posiciona os elementos na tela. Existem várias formas de criar Layouts com CSS:

- Fluxo normal
- Float
- Position
- Flexbox
- Grid

LAYOUT | FLUXO DE RENDERIZAÇÃO

I am a basic block level element. My adjacent block level elements sit on new lines below me.

By default we span 100% of the width of our parent element, and we are as tall as our child content. Our total width and height is our content + padding + border width/height.

We are separated by our margins. Because of margin collapsing, we are separated by the width of one of our margins, not both.

inline elements like this one and this one sit on the same line as one another, and adjacent text nodes, if there is space on the same line. Overflowing inline elements will wrap onto a new line if possible (like this one containing text), or just go on to a new line if not, much like this image will do:

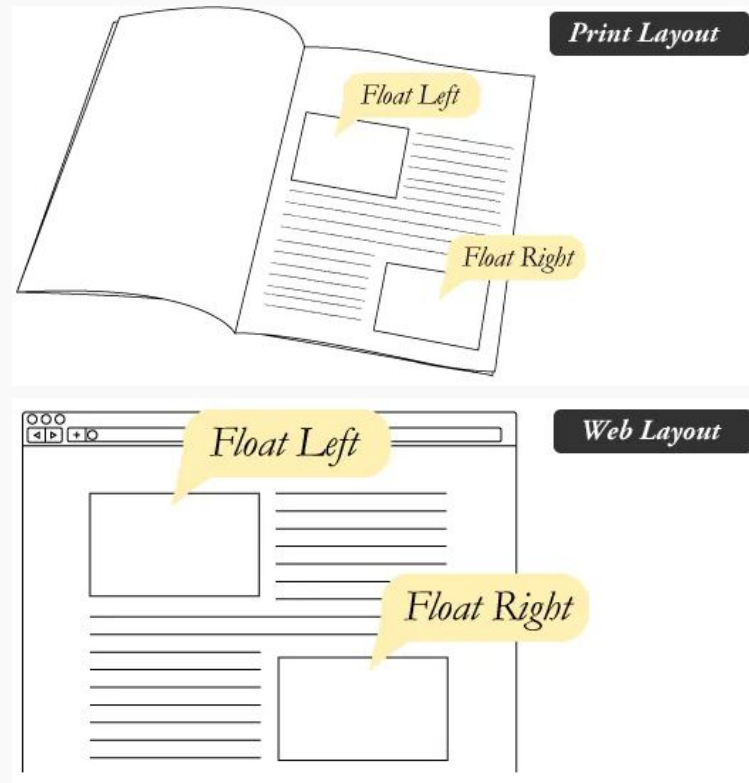


display: block

display: inline

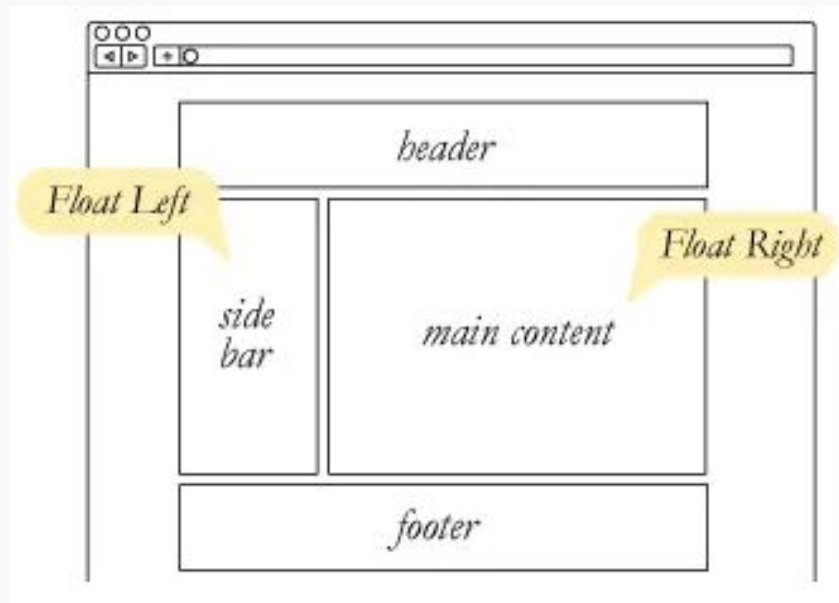
LAYOUT | FLOAT

- Originalmente criado para simular layouts de jornais, onde textos "contornam" imagens.
- Retira o elemento de seu fluxo normal e coloca ao lado direito/esquerdo dentro do seu contêiner, ajustando textos e elementos inline ao seu redor.

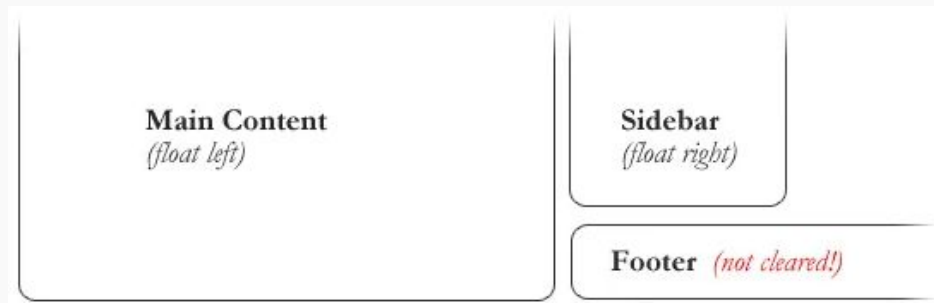


LAYOUT | FLOAT

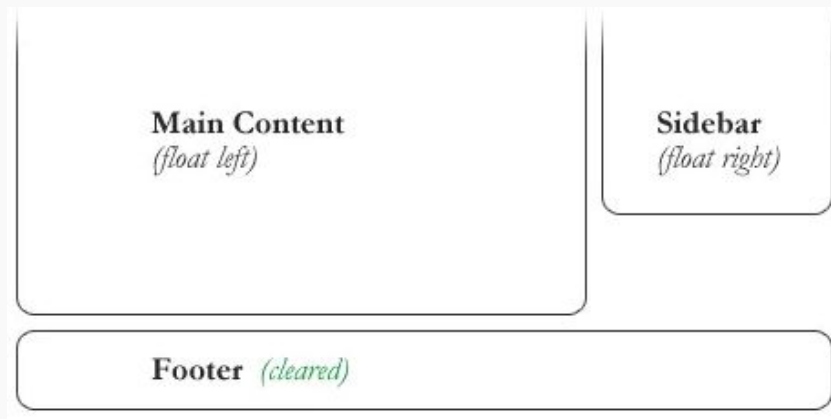
- Existem 4 valores para a propriedade float:
 - left
 - right
 - none
 - inherit
- Também pode ser utilizado para criar layouts inteiros.



- Clear: propriedade irmã do float.
- Um elemento com a propriedade clear definida não subirá adjacente ao elemento float, mas se moverá além do elemento float.



- Também existem 4 valores para a propriedade clear:
 - both
 - right
 - left
 - none
- Aplicando a propriedade "both" no exemplo anterior:



- Ainda muito utilizado para Layouts, porém existem ferramentas mais poderosas para criar Layouts CSS:
 - Position
 - Flexbox
 - Grid

CSS POSITION

Manipula o fluxo normal de renderização do documento, deslocando elementos de acordo com condições definidas:

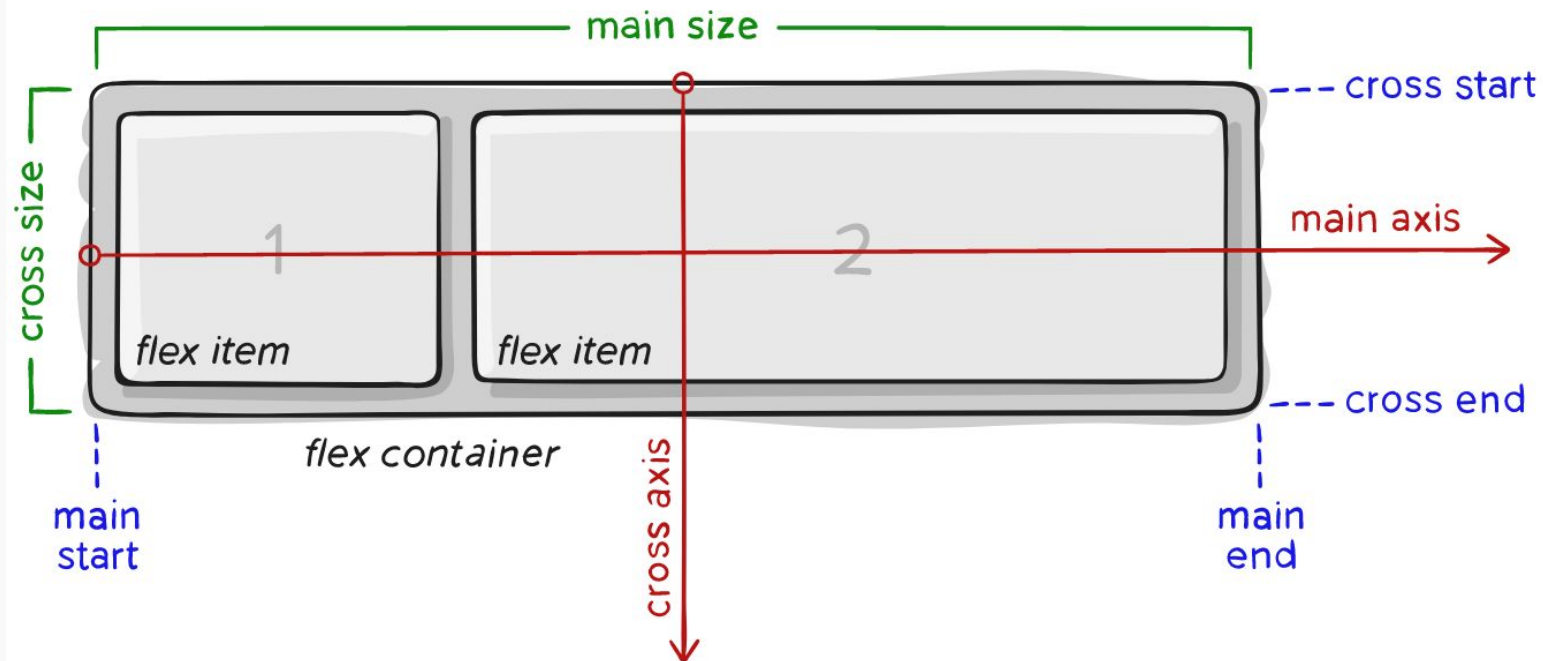
- **static**: padrão, segue o fluxo normal.
- **absolute**: retira o elemento do fluxo normal e o posiciona em relação ao ancestral mais próximo. Define a distância para esse ancestral através dos atributos *top*, *right*, *bottom* e *left*.
- **relative**: mantém o elemento no fluxo normal, mas se utilizarmos os atributos *top*, *right*, *bottom* ou *left*, posiciona o elemento relativamente à sua posição original (que tem seu espaço preservado no documento).

CSS POSITION

- **fixed**: remove o elemento do fluxo normal e o posiciona relativamente ao contêiner inicial do documento (*viewport*), mantendo o elemento sempre visível, fixo nessa posição, mesmo com rolagem de tela.
- **sticky**: mantém o elemento no fluxo normal, preservando seu espaço no documento. Mas ao rolar a tela, o elemento "gruda" no *viewport* e se mantém sempre visível (respeitando as distâncias *top/right/bottom/left*).

```
position: absolute;
```

FLEXBOX E LAYOUTS MODERNOS



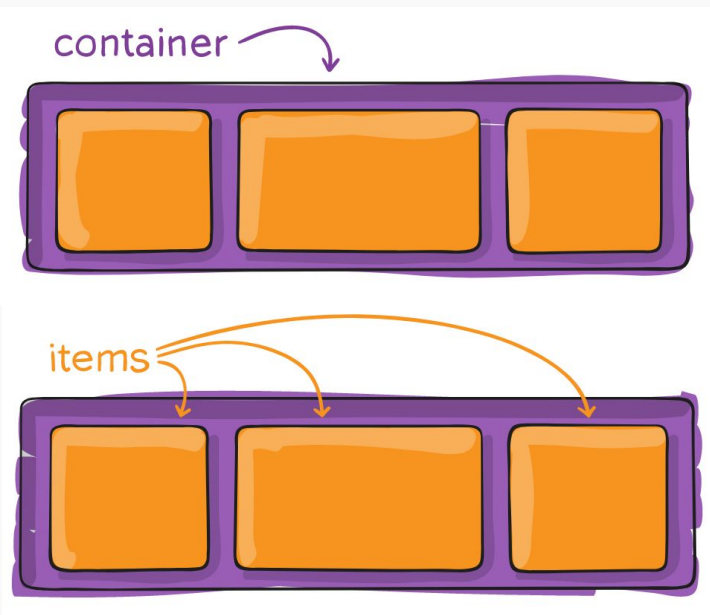
FLEXBOX CONTAINER | DISPLAY

Quando utilizamos flexbox nos referimos a 2 elementos básicos.

Algumas propriedades são aplicadas ao **container**, outras aos **items**

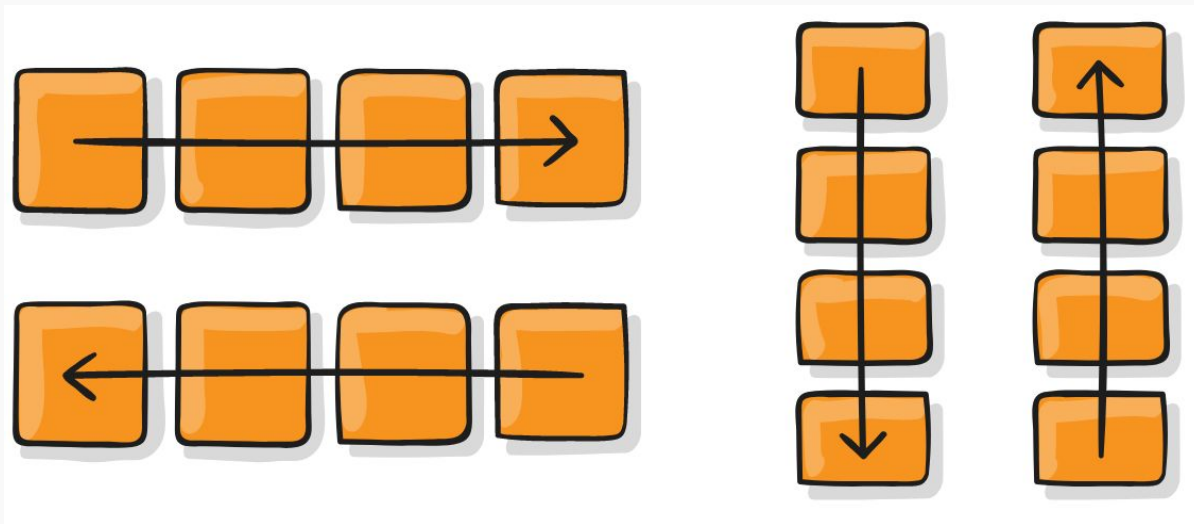
```
.container {  
  display: flex;  
} /* ou inline-flex */
```

```
<div class="container">  
  <p>Lorem.</p>  
  <p>Ipsum.</p>  
  <p>Dolor.</p>  
</div>
```



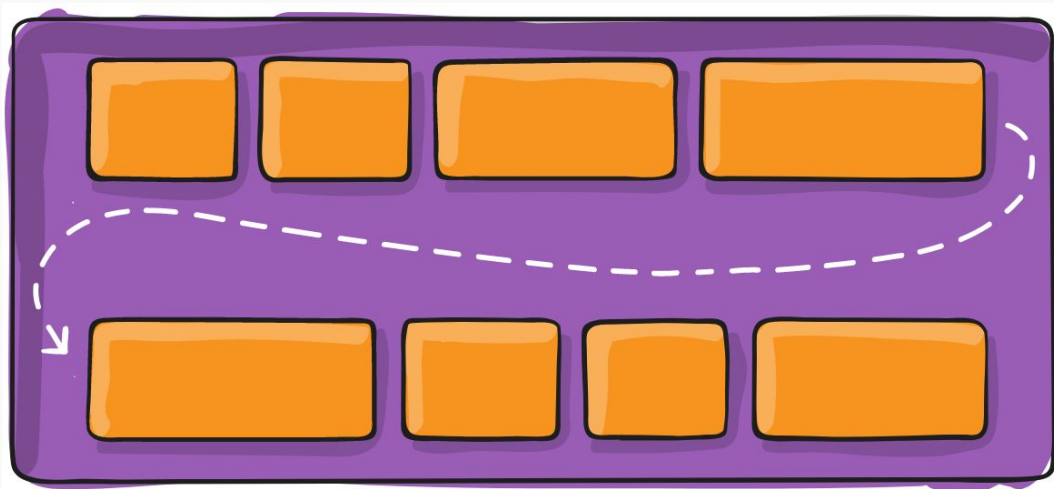
FLEXBOX CONTAINER | FLEX-DIRECTION

```
flex-direction: row | row-reverse | column | column-reverse;
```



FLEXBOX CONTAINER | FLEX-WRAP

```
flex-wrap: nowrap | wrap | wrap-reverse;
```



A propriedade **flex-wrap** define um comportamento interno do container

O efeito é que na organização dos seus **itens**, quando necessário a linha poderá ou não ser interrompida.

Chamamos isso de quebra de linha.

a propriedade **nowrap** é padrão, a propriedade **wrap-reverse** faz com que a linha ao terminar continue no sentido reverso.

FLEXBOX CONTAINER | JUSTIFY-CONTENT

`justify-content:`

flex-start



flex-end



center



space-between



space-around



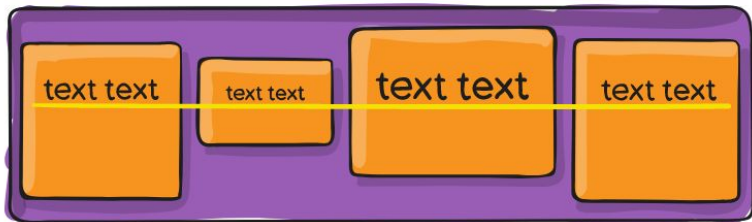
space-evenly



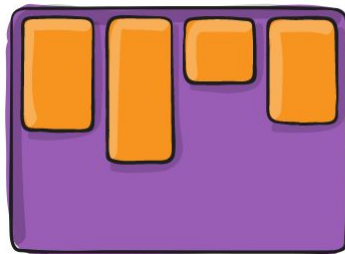
FLEXBOX CONTAINER | ALIGN-ITEMS

`align-items:`

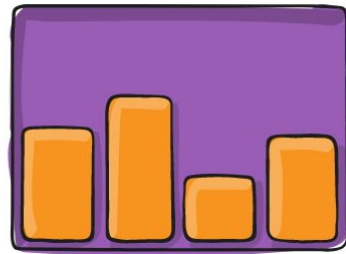
baseline



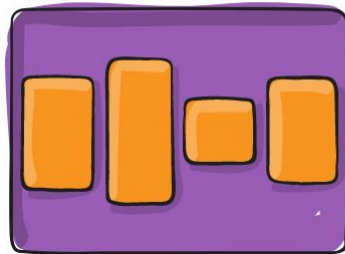
flex-start



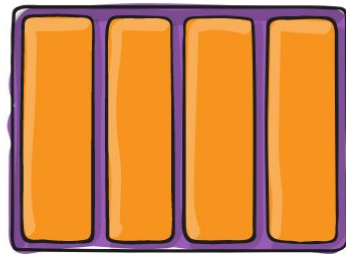
flex-end



center



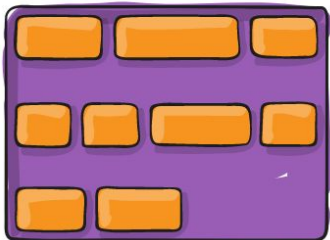
stretch



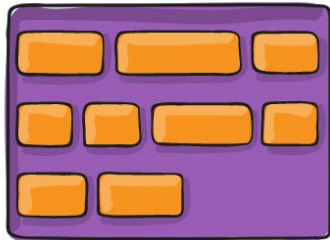
FLEXBOX CONTAINER | ALIGN-CONTENT

`align-content:`

space-between

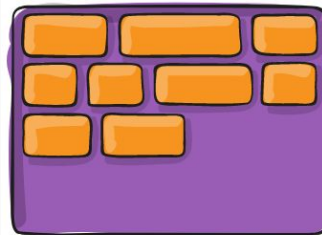


space-around

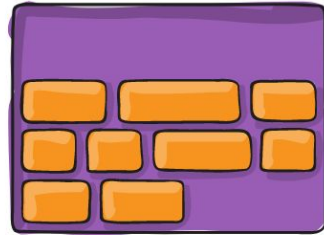


para flex-wrap: wrap ou wrap-reversed

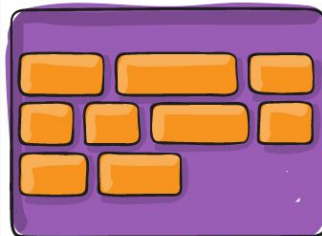
flex-start



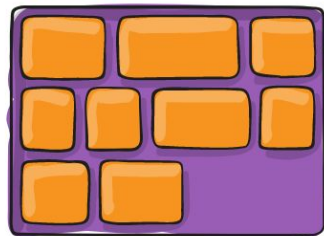
flex-end



center



stretch



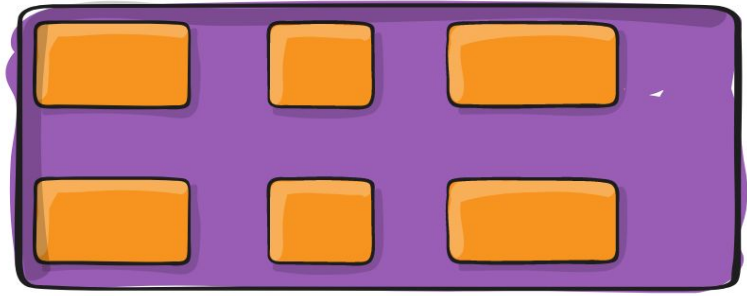
FLEXBOX CONTAINER | GAP

```
gap: | row-gap: | column-gap:
```

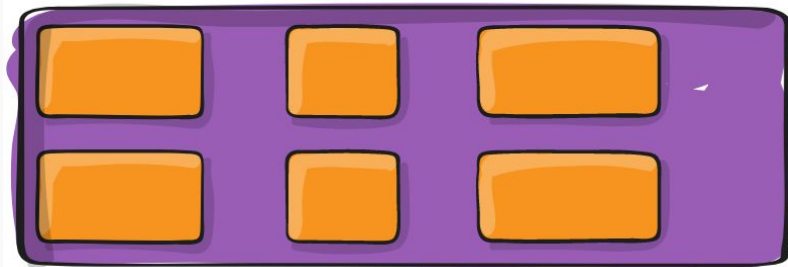
gap: 10px



gap: 30px



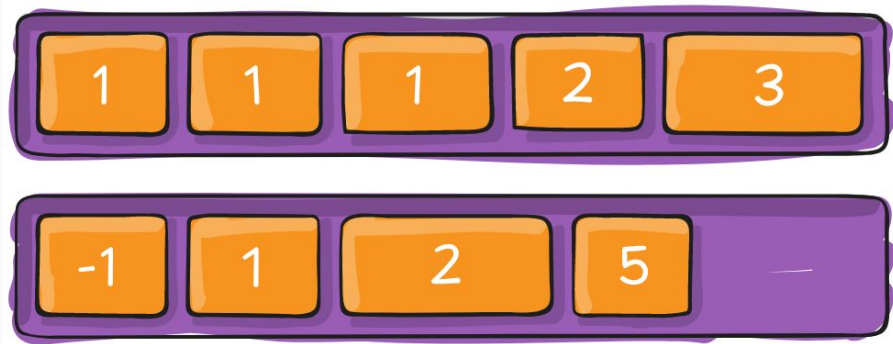
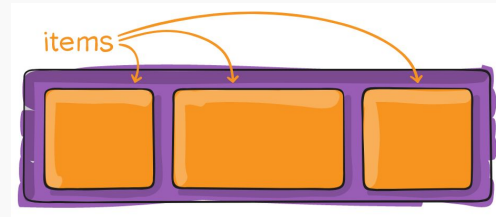
gap: 10px 30px



Define de forma explícita a distância (margin) entre os itens flexíveis.

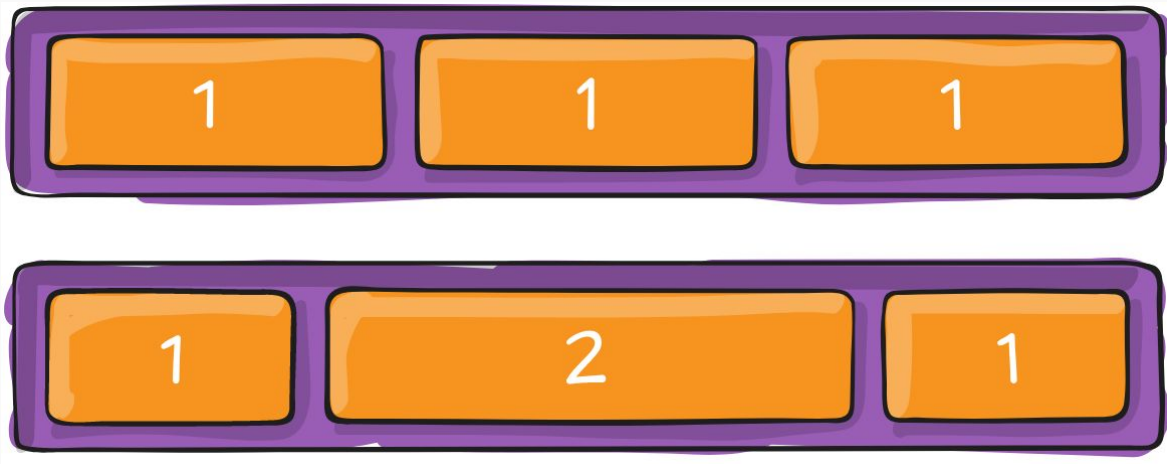
FLEXBOX ITEM | ORDER

```
order: 3
```



FLEXBOX ITEM | FLEX

```
flex-grow: 1 | flex-shrink: 1 | flex-basis: 100px
```

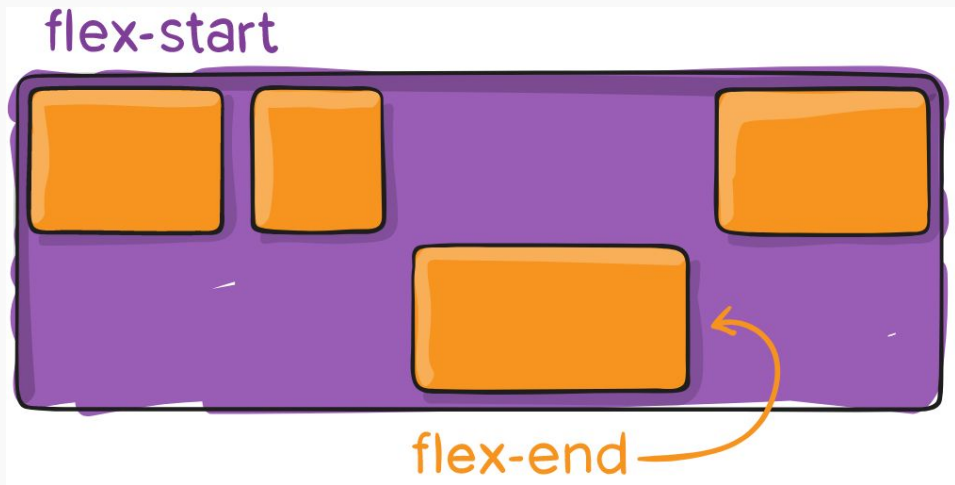


```
flex: 1 1 100px;
```

A propriedade **flex-grow** habilita um **item crescer quando necessário**.
Com **valores numéricos**, Os itens flexíveis irão se ajustar proporcionalmente

FLEXBOX ITEM | FLEX

```
align-self: auto | flex-start | flex-end  
           center | baseline | stretch
```



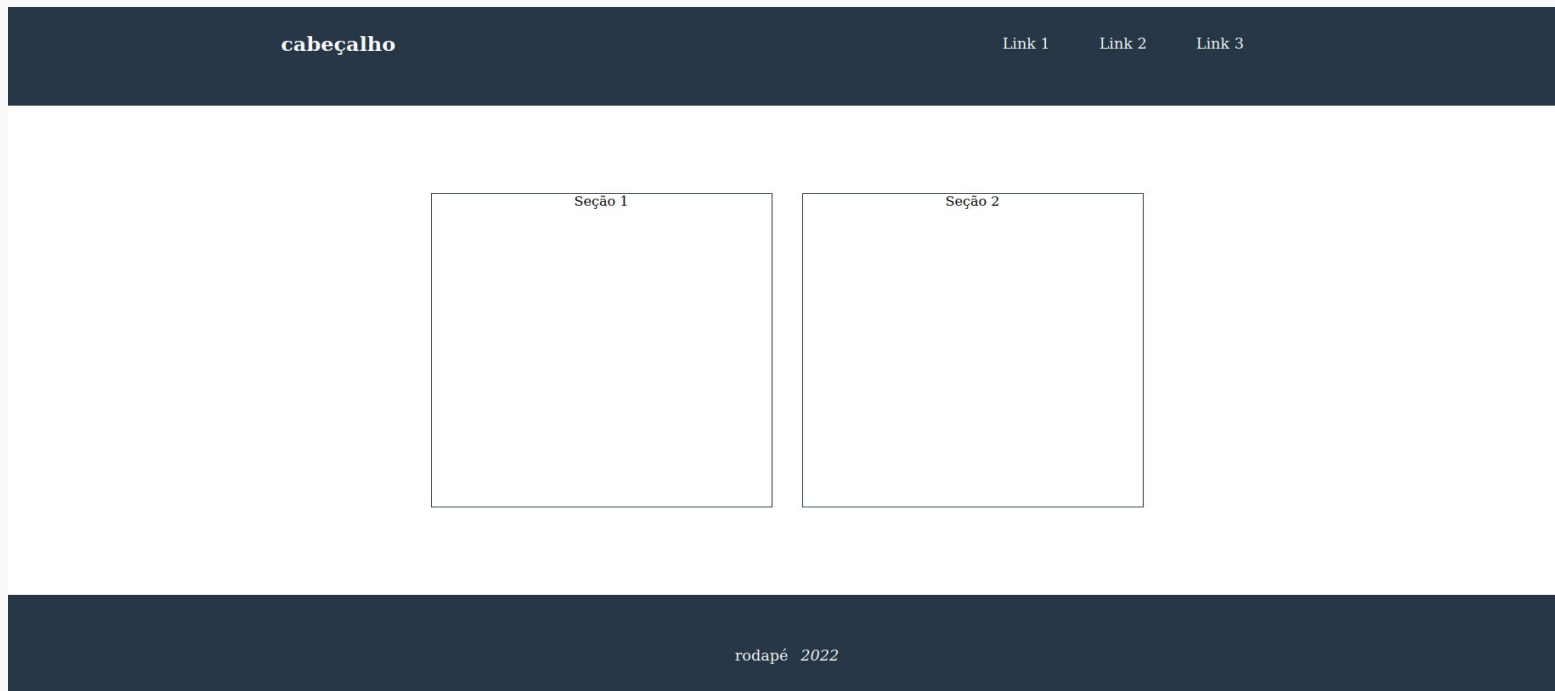
CSS MEDIA QUERY

- **Media Queries** são úteis quando se deseja modificar seu *site/app* de acordo com o tipo (***all, print, screen***) ou alguma característica específica (por exemplo, largura da tela).
- Se os parâmetros definidos na *query* forem verdadeiros, o navegador aplica os estilos definidos.

```
@media screen and (max-width: 800px) {  
  .right {  
    width: 100%;  
  }  
}
```

PRÁTICA COM DESAFIO

- Construir página com o layout apresentado na imagem abaixo:



MATERIAL COMPLEMENTAR



HTML Full Course | <https://youtu.be/pQN-pnXPaVg>

CSS3: Box Model | <https://youtu.be/n2sp9tgEdag>

Learn Every CSS Selector In 20 Minutes | <https://youtu.be/l1mER1bV0N0>

CSS: Seletores e Especificidade | <https://youtu.be/dPL23aVRllc>

Entendendo sobre position no CSS | <https://youtu.be/Y7NegpwLM2g>

Learn CSS Position In 9 Minutes | <https://youtu.be/jx5jml0UIXU>

Flexbox CSS In 20 Minutes | <https://youtu.be/IISoEo8ISnc>

CSS Flexbox in 100 Seconds | <https://youtu.be/K74l26pE4YA>

Flexbox design patterns you can use in your projects | <https://youtu.be/vQAvjof1oe4>

Aprenda Flexbox em 20 minutos | <https://youtu.be/P9TrFDNwor4>

Como utilizar Media Queries para sites Responsivos | <https://youtu.be/AltqAPZzAqo>

MATERIAL COMPLEMENTAR



Box model - CSS | developer.mozilla.org/docs/Web/CSS/CSS_Box_Model/Introduction_to_the_CSS_box_model

Seletores CSS | https://developer.mozilla.org/pt-BR/docs/Web/CSS/CSS_Selectors

Pseudo-classes | <https://developer.mozilla.org/pt-BR/docs/Web/CSS/Pseudo-classes>

Pseudo-elementos | https://developer.mozilla.org/pt-BR/docs/Web/CSS/CSS_Selectors#pseudo-elementos

:nth-child() | <https://developer.mozilla.org/pt-BR/docs/Web/CSS/:nth-child>

Specificity Calculator | <https://specificity.keegan.st> (calculadora de especificidade)

CSS Layout | https://developer.mozilla.org/en-US/docs/Learn/CSS/CSS_layout

Conceitos básicos de flexbox | https://developer.mozilla.org/pt-BR/docs/Web/CSS/CSS_Flexible_Box_Layout/Basic_Concepts_of_Flexbox

Guia completo de Flexbox | <https://origamid.com/projetos/flexbox-guia-completo>

A Complete Guide to Flexbox | <https://css-tricks.com/snippets/css/a-guide-to-flexbox>

Position CSS | <https://developer.mozilla.org/pt-BR/docs/Web/CSS/position>

CSS @media Rule | https://www.w3schools.com/cssref/css3_pr_mediaquery.asp

Grid Layout | <https://css-tricks.com/snippets/css/complete-guide-grid>

Grid Layout w3schools | https://www.w3schools.com/css/css_grid.asp

AVALIAÇÃO DOCENTE

O que você está achando das minhas aulas neste conteúdo?

[Clique aqui](#) ou escaneie o QRCode ao lado para avaliar minha aula.

Sinta-se à vontade para fornecer uma avaliação sempre que achar necessário.





DEVinHouse

Parcerias para desenvolver a sua carreira

OBRIGADO!



<LAB365>