

# Iniciação à Programação com: HTML, CSS e Java Script

Projeto Beira Linha



PUC Minas

# Quem somos nós ?



**Daniel Victor Rocha Costa**

dani.vitorcosta@gmail.com  
(31) 986645625



**Fabia Rodrigues de Oliveira**

fabia9065@gmail.com  
(31) 996391141



**Gabriel Ferreira Pereira**

gabrielferreirape502@gmail.com  
(31) 972471784

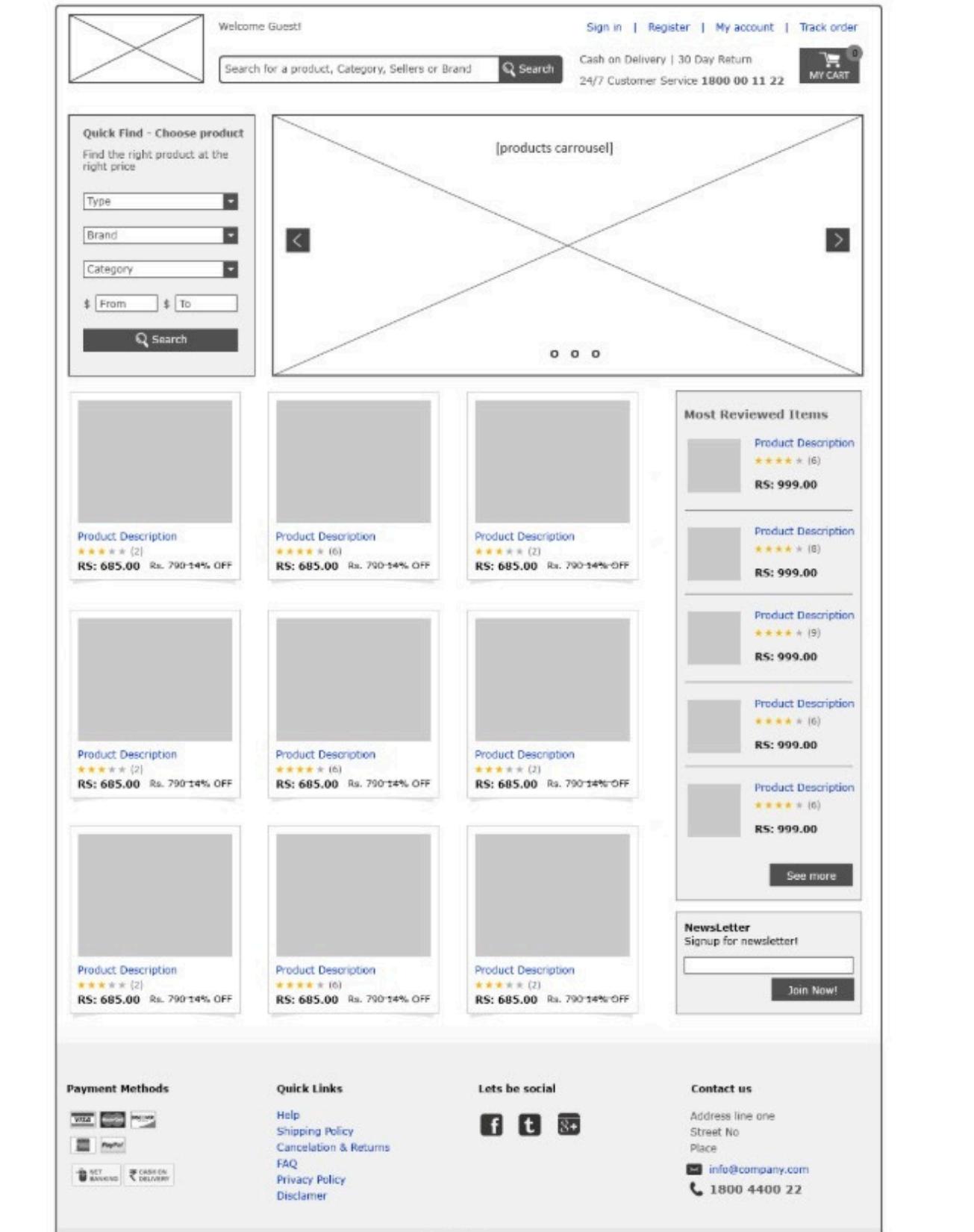


# Trabalho prático 1: Construindo seu Primeiro E-commerce

# Escolha um dos temas a seguir

1. Roupas
2. Eletrônicos
3. Brinquedos
4. Papelaria
5. Instrumentos musicais
6. Produtos para cabelo
7. Comida

# Esboço e-commerce



# Vamos iniciar trabalhando com o header (topo) da nossa página

The image shows the top navigation bar of a food delivery application. It features a red header bar with rounded corners. On the left is a circular profile picture placeholder containing a bowl of pizza. To its right is a search bar with a magnifying glass icon and the text "Pesquisar". Further right are three icons: a star for "Favoritos", a shopping cart for "Carrinho", and a user icon for "Perfil". Below the header is a white content area with a red border. The word "Categorias" is displayed in red text. Below it are four rounded buttons labeled "Todas", "Vegana", "Doce", and "Combos", each in red text.

Categorias

Todas

Vegana

Doce

Combos

Pesquisar

Favoritos

Carrinho

Perfil

# <header> </header>

```
<header>
```

```
    | <p> Topo da página </p>
```

```
</header>
```

# <nav> </nav>

```
<nav>
  <ul>
    <li> Favoritos </li>
    <li> Carrinho </li>
    <li> Perfil </li>
  </ul>
</nav>
```

# Mão na massa!

Vamos fornecer uma  
página tutorial para  
auxiliar na construção do  
header.

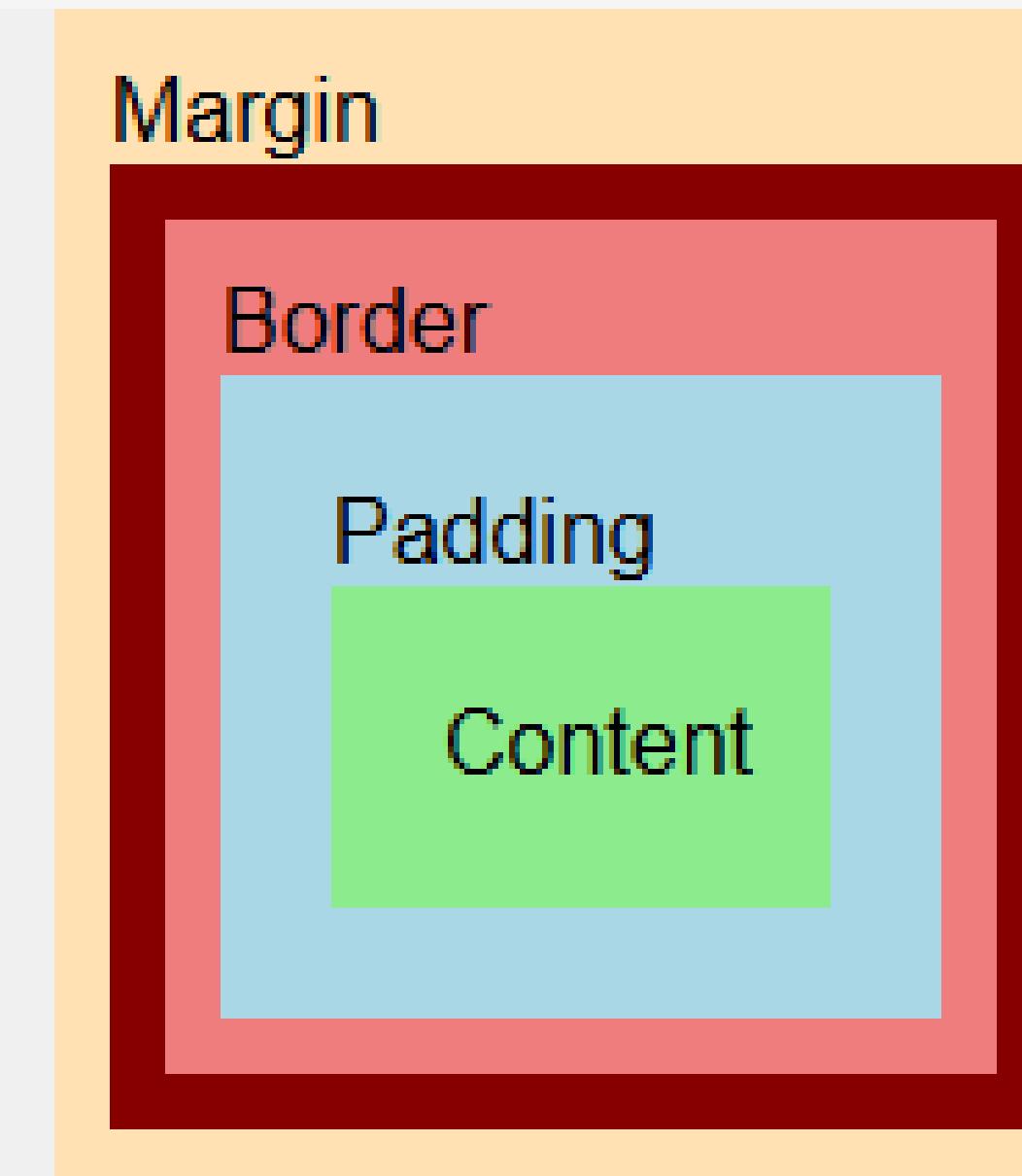
**Para adicionar imagem: Acesse o site  
FREEPIK**

**Para combinação de cores: Acesse o site:  
MY BRAND NEW LOGO**



# Box model

Todo elemento HTML é uma caixa com 4 partes:





Pai Filho

A caixa roxa representa o elemento pai.

Enquanto a caixa verde representa o elemento filho.

**Podemos perceber que o filho está colado no elemento pai.**

**Dessa forma, vamos adicionar um espaçamento interno na caixa do filho (padding).**



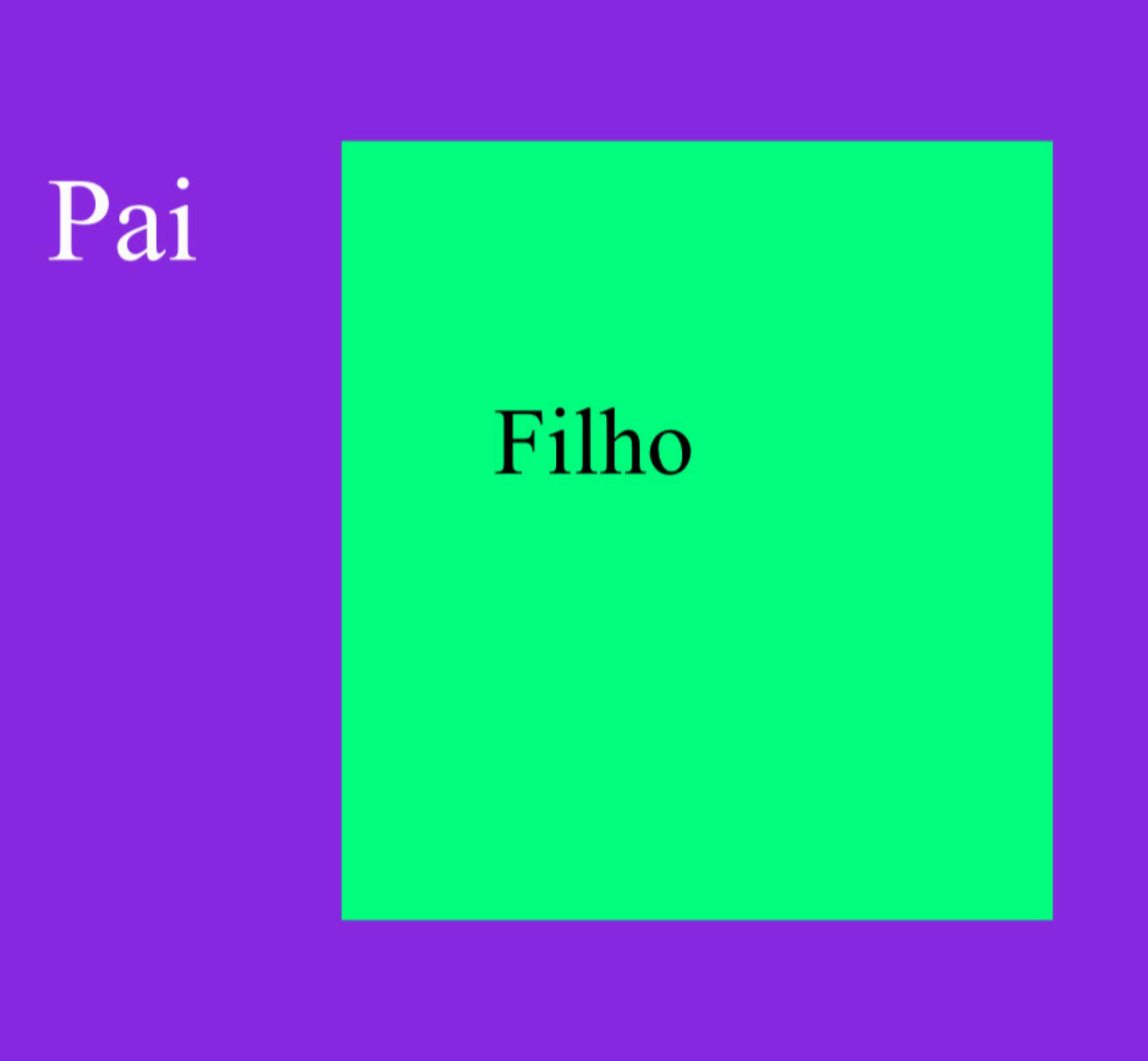
Pai

Filho

```
.filho {  
    padding: 2em;  
}
```

**Agora, com o preenchimento interno,  
o meu elemento filho aumentou o  
tamanho do seu conteúdo.**

**Para que a gente consiga separar o  
elemento filho do elemento pai,  
vamos adicionar o espaçamento  
externo (margem).**



Pai

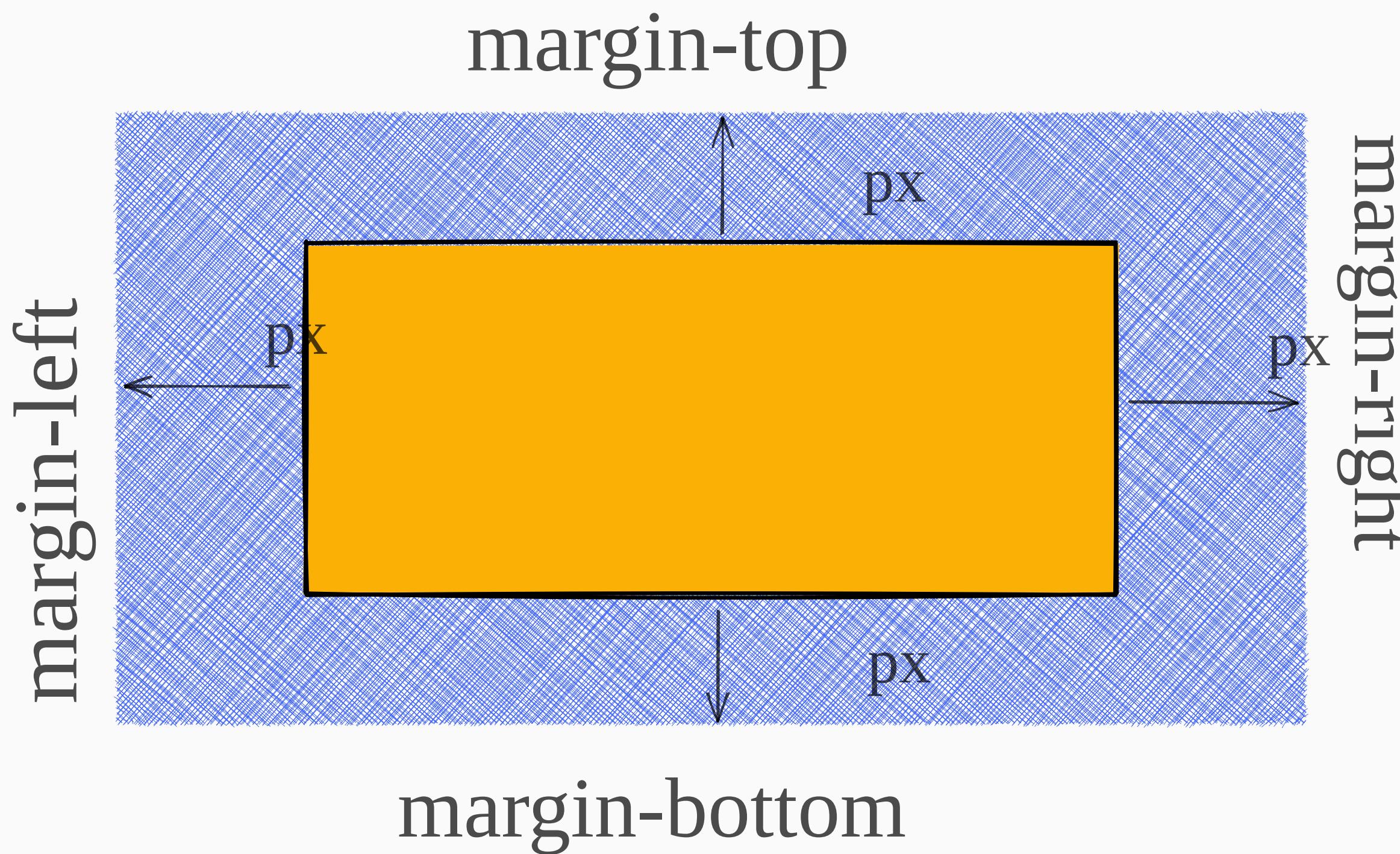
Filho

```
.filho {  
    margin: 2em;  
}
```

**Agora o meu elemento filho possui tanto um espaçamento interno quanto um espaçamento externo.**

**Por fim, vamos adicionar uma borda, ou seja, o contorno para o nosso elemento filho.**

margin: px;



Pai

Filho

**.filho {**

**border: 8px solid white;**

**}**



# FlexBox

# Display

Utilizo essa propriedade para definir como os elementos serão organizados na página – se ficarão em linha (um ao lado do outro) ou em coluna (um abaixo do outro).

Além disso, ela permite criar um posicionamento flexível, facilitando o alinhamento e a distribuição dos elementos de diferentes maneiras na interface.

# Flex

A propriedade que me permite organizar os elementos filhos de um elemento pai é o Flex.

Antes de aplicá-la no código, é importante identificar e nomear o elemento pai, pois é nele que adicionamos a propriedade **display: flex;**.

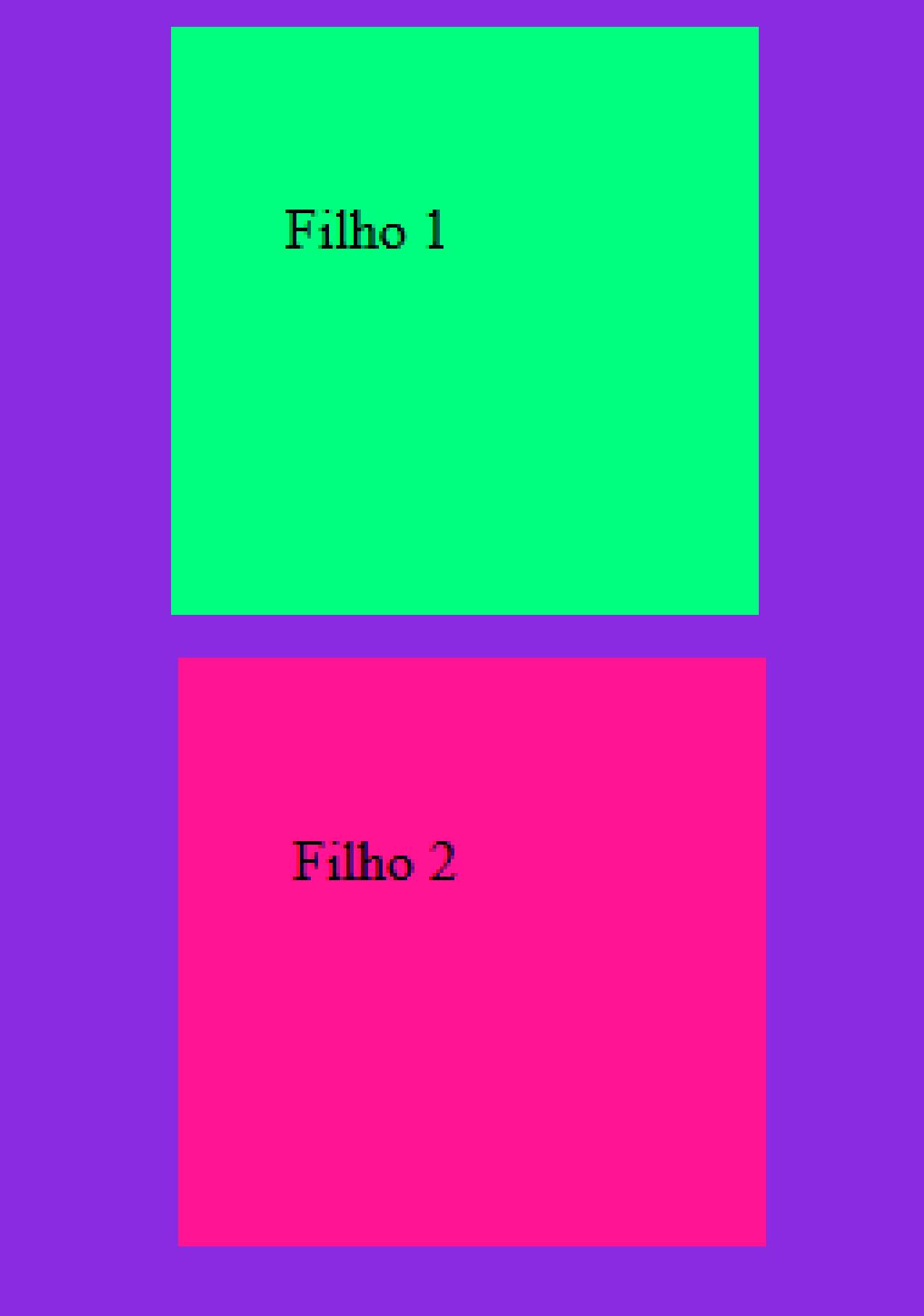
```
<div class = "pai">  
    <div class = "filho">  
        <p> Elemento Filho </p>  
    </div>  
</div>
```

Nesse exemplo, vocês conseguem identificar quem é o elemento pai, e quem é o elemento filho?

# E nesse exemplo?

```
<div class = "pai">  
    <div class = "filho1">  
        <p> Filho 1 </p>  
  
    <div class = "filho2">  
        <p> Filho 2 </p>  
    </div>  
  
</div>
```

**Com o elemento pai e os elementos filhos identificados, podemos iniciar a estilização.**

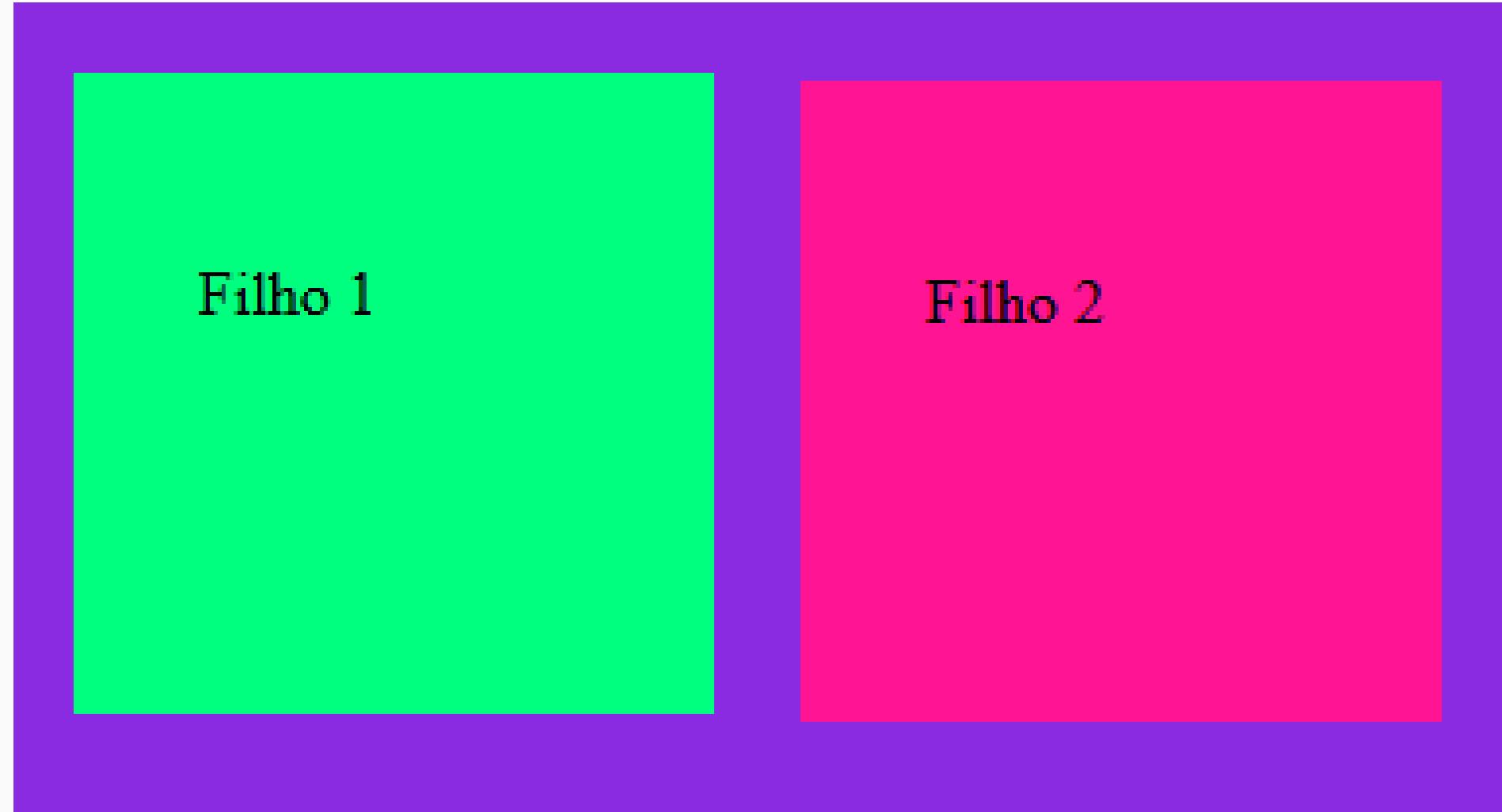


Filho 1

Filho 2

Vamos deixar esses elementos um ao lado do outro.

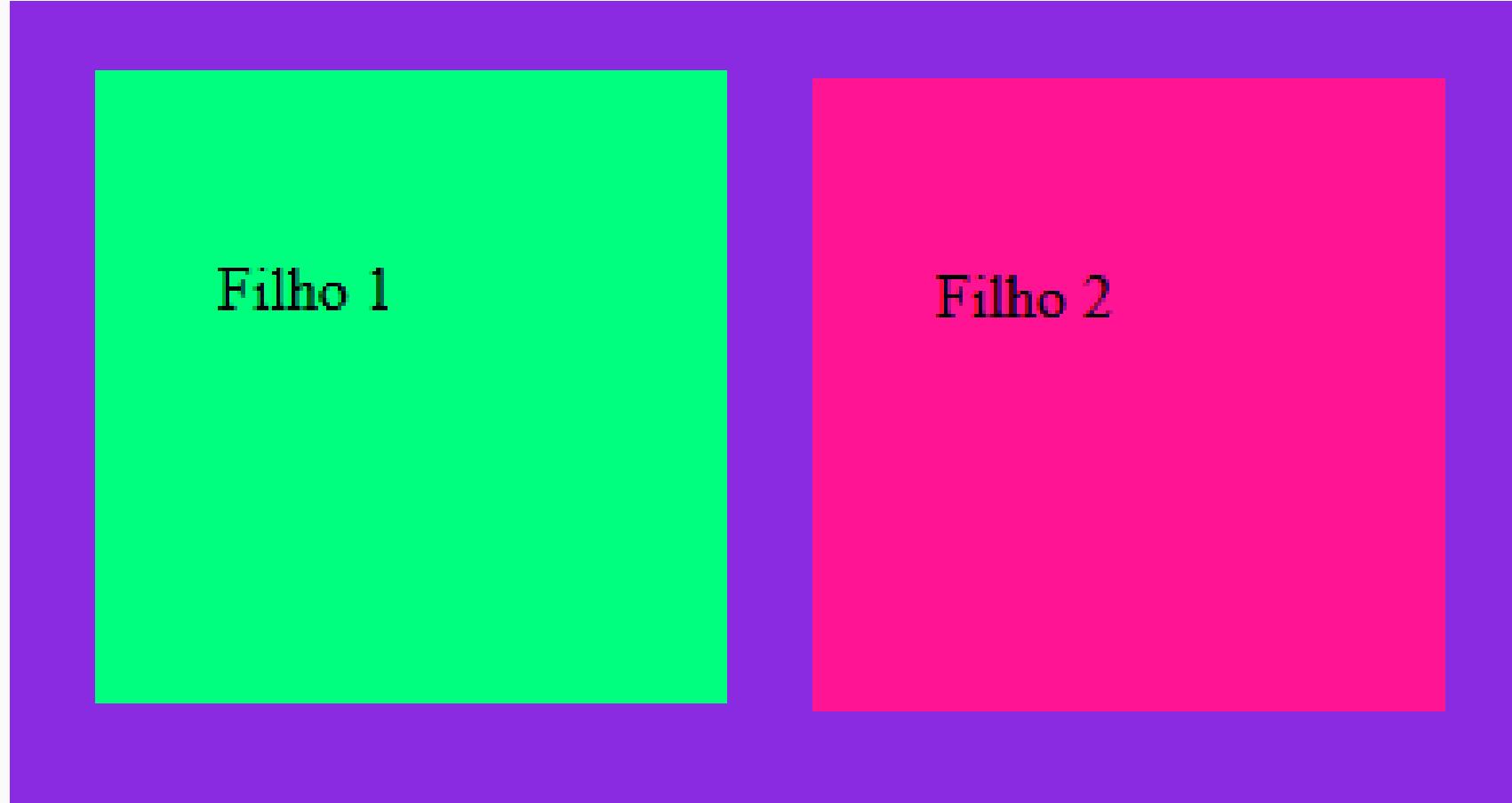
O elemento pai é representado pela cor roxa.



```
.elemento-pai {  
    display: flex;  
}  
}
```

**Dessa forma, coloco as caixas uma ao lado da outra.**

**Todavia, se o posicionamento não mudar, podemos adicionar mais uma propriedade.**

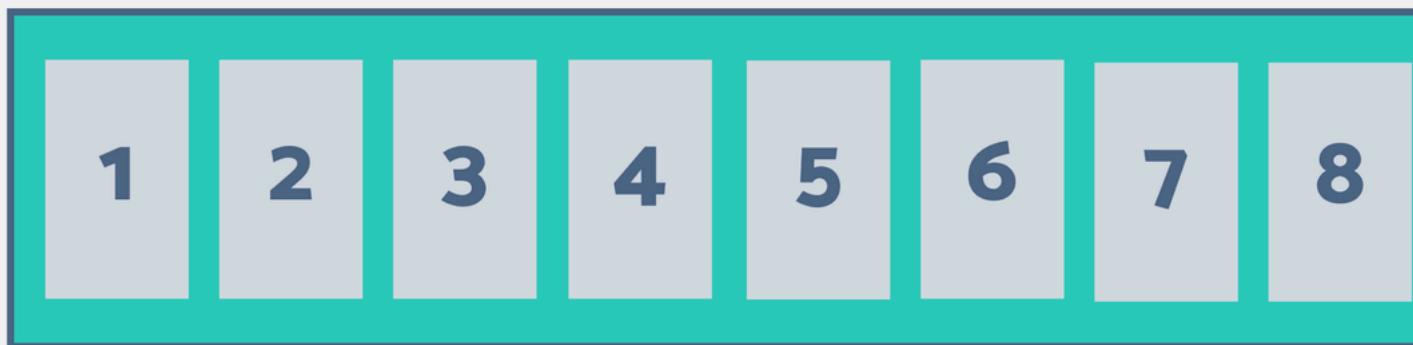


```
.elemento-pai {  
    display: flex;  
    flex-direction: row;  
}
```

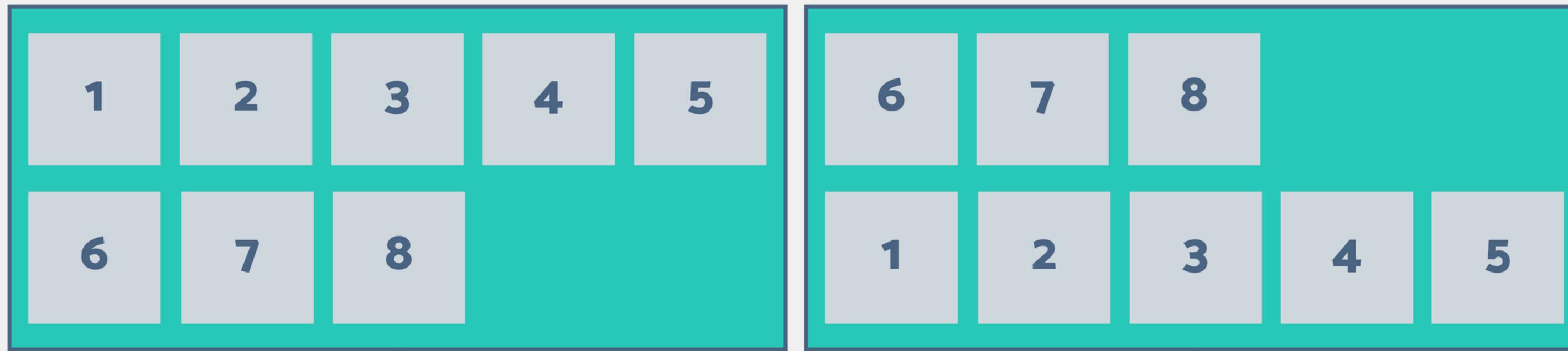
**Para centralizar os meus itens, vou aplicar uma terceira propriedade.**

```
.elemento-pai {  
    display: flex;  
    flex-direction: row;  
    justify-content: center;  
}
```

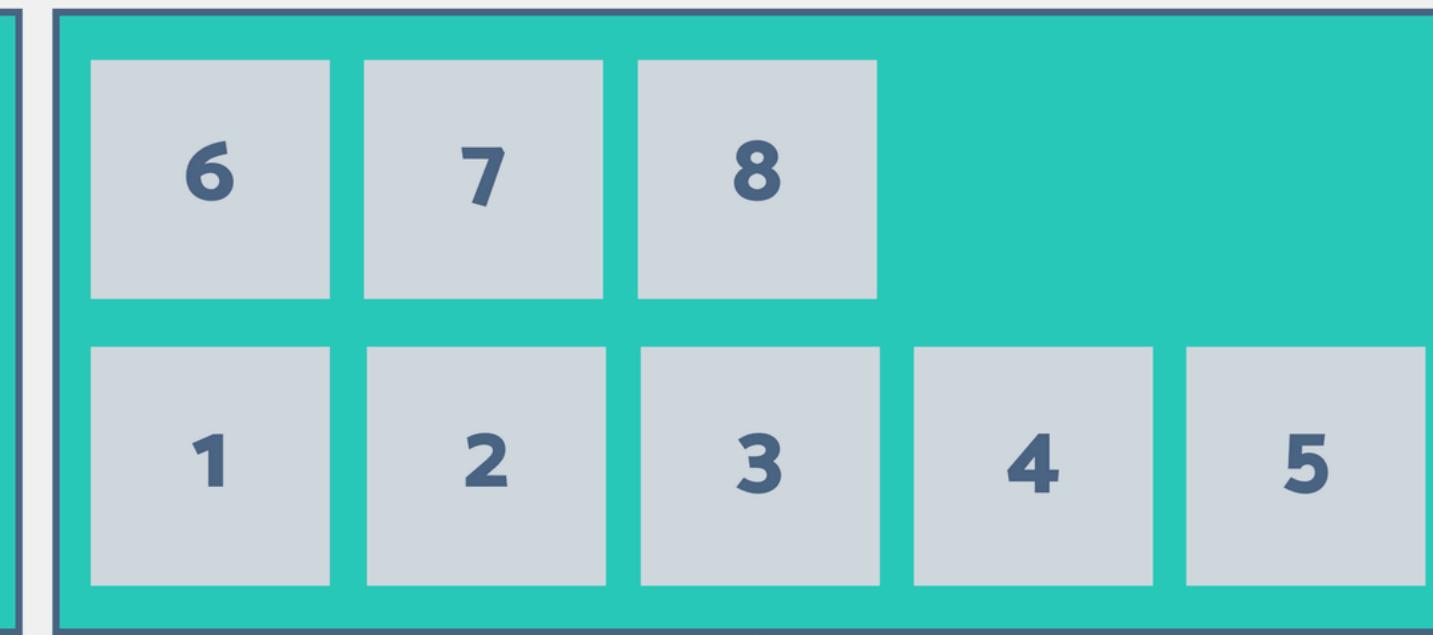
# flex-wrap



flex-wrap: nowrap

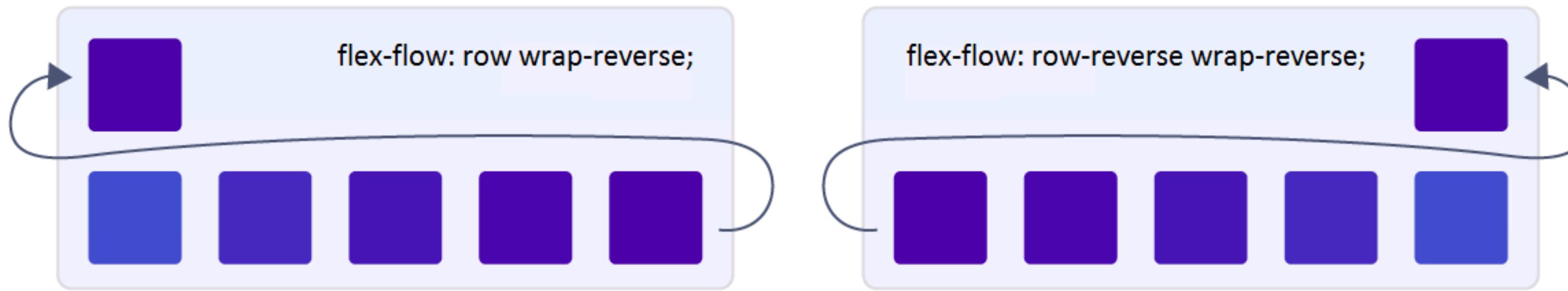
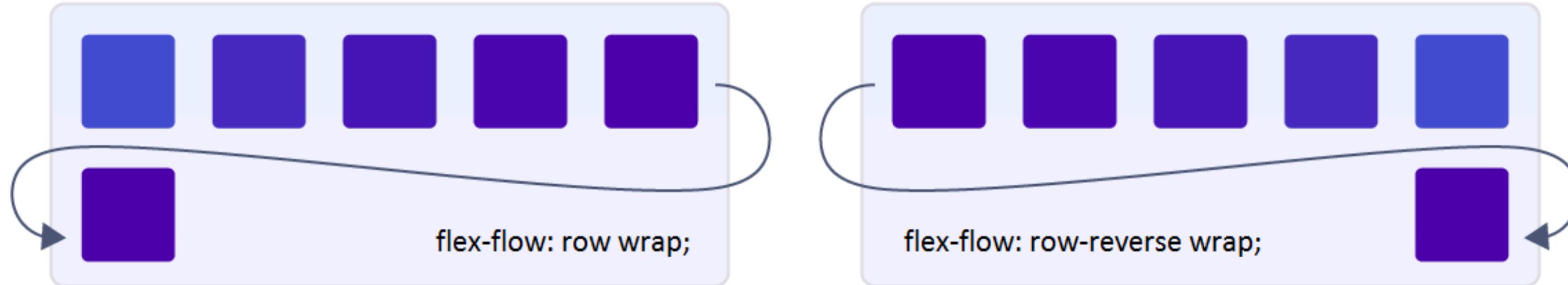


flex-wrap: wrap



flex-wrap: wrap-reverse

## Flexbox result:



## Desired result:



**Dessa forma, finalizamos a  
formatação do posicionamento do  
novo header**

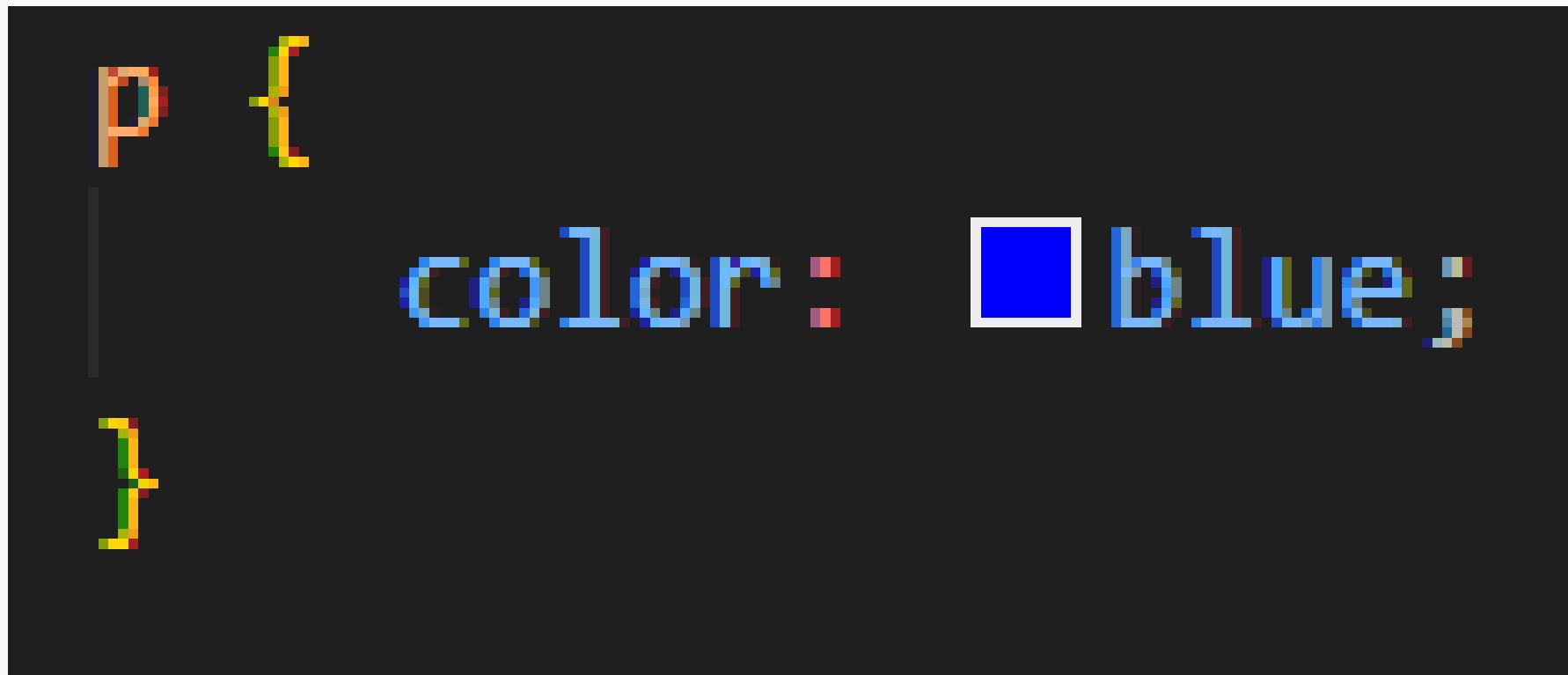


Dúvidas?

# Tipos de seletores

# O que são seletores?

Seletores definem quais elementos HTML o CSS vai estilizar



```
p {  
    color: blue;  
}
```

Todos os parágrafos terão o texto azul



# O que são Classes e Ids ?

- Servem para **identificar** elementos no HTML. Como se dêssemos um apelido ao elemento para chama-lo no CSS.
- São usados principalmente para aplicar estilos no **CSS** ou acessar elementos com **JavaScript**.
- Funcionam como “**rótulos**” que você coloca em uma tag.



# Classes

- Usada para agrupar elementos que **compartilham o mesmo estilo**.
- Pode ser usada em vários elementos ao mesmo tempo.
- Definidas por **class="nomeDaClasse"** dentro da tag desejada.
- É referenciada com ponto (.) no CSS

```
<div class="city">
    <h2>Londres</h2>
    <p>Londres é a capital da Inglaterra.</p>
</div>

<div class="city">
    <h2>Paris</h2>
    <p>Paris é a capital da França.</p>
</div>

<div class="city">
    <h2>Tóquio</h2>
    <p>Tóquio é a capital do Japão.</p>
</div>
```



# Id

- Usado para identificar um **único** elemento específico na página.
- Cada ID deve ser único (não se repete).
- É escrito com # (cerquilha) no CSS.
- Definido por **id="MeuId"** dentro de uma tag

```
<h1 id="minhaHeader">Header</h1>

<div>
    <p id="meuParagrafo">bla bla bla</p>
</div>
```

# Tipos de seletores

**Universal (\*)** - aplica o estilo a todos os elementos da página.

```
* {  
    margin: 0;  
    padding: 0;  
}
```

**De tipo** (ex: h1) - seleciona todas as tags de um mesmo tipo.

```
p {  
    color: red;  
}
```

**classe (.)** – seleciona elementos com uma classe específica.

```
.destaque {  
    font-weight: bold;  
}
```

**ID (#)** – seleciona um elemento único que tenha um identificador.

```
#principal {  
    background-color: #lightgray;  
}
```

De **atributo** – seleciona elementos com um atributo específico.

```
[type="text"] {  
    border: 2px solid #blue;  
}
```

# Exemplo

## Html

```
<p class="info">Texto 1</p>
<p id="principal">Texto 2</p>
<input type="text">
```

## CSS

```
.info {
    color: green;
}

#principal {
    font-weight: bold;
}

[type="text"] {
    border: 2px solid blue;
}
```

# Combinação de Seletores

## Descendente

div p

Seleciona todos os <p> dentro de <div>

```
div p {  
    color: yellow;  
}
```

## Filho direto

div > p

Seleciona apenas os <p> que são filhos diretos da <div>

```
div > p {  
    color: red;  
}
```

# Combinação de Seletores

## Múltiplos

h1, h2, h3

Aplica o mesmo estilo a todos os seletores listados

```
h1 h2 h3 {  
    color: green;  
}
```

## Irmão geral

h1 ~ p

Seleciona todos os <p> que vêm depois de um <h1>

```
h1 ~ p {  
    color: white;  
}
```



Dica de estudo !

# Mimo



Aplicativo interativo para aprender programação.



Estudar em qualquer lugar, direto no celular.



Conteúdo em formato gamificado, que torna o aprendizado mais divertido.

Disponível para Android e iOS.

