

Curso: Front-End do Bem



AULA 09

- O que é FlexBox?
- Propriedades do FlexBox
- Exemplos de FlexBox
- Sugestões de Prática para essa aula
 - Links das 8 aulas anteriores
- O que vamos aprender na aula 10



O que é FlexBox?

FlexBox é uma solução de layout para conseguirmos alinhar e distribuir itens dentro de um container. Quando o tamanho da página é desconhecido ou quando ele é dinâmico e isso reflete na forma como desenvolvemos aplicações para web hoje em dia.

Porque se fomos parar pra pensar (um site hoje em dia) ele é acessado não só do desktop, mas também do smartphone, do tablet e por ai vai. Todos esses dispositivos eles tem resoluções e tamanhos de tela diferente ou seja: quando vamos desenvolver um site temos que pensar que a pessoa que vai acessar pode ter um smartphone com a tela pequenininha ou até aquelas Tv's de 85 polegadas.

Então quando vamos criar um site, precisamos pensar no layout dele de uma forma que ele seja facilmente adaptado (e esse conceito tem um nome, um não sei se eu falo agora ou deixo para as próximas aulas, esse conceito de adaptar o layout do site para tamanhos de telas diferentes é chamado de Responsivo) ops falei!!! A ideia é deixar esse conceito de Responsivo plantado na cabeça de vocês (vamos ver esse conceito em uma aula mais pra frente um pouquinho desse curso).

Resumindo:

O FlexBox cria containers e itens dentro desses containers flexíveis que se ajustam a essas diferentes resoluções de tela.



O que é FlexBox?

A partir de agora vamos entender sobre as propriedades do FlexBox:

- Aquelas que usamos nos containers;
- Aquelas que usamos dentro dos itens que estão dentro desses containers

Se fizermos aquela analogia marota para a nossa casa, ficaria assim:

- A parede seria o container;
- E os revestimentos (tijolinho) os itens dentro desse container(parede).



Vamos entender isso melhor com os conceitos dessas propriedades e exemplos já na próxima página dessa aula.

Partiu.....

Propriedades do FlexBox

Para começarmos a utilizar o Flexbox vamos precisar de um container e o elemento que vamos usar será uma `<div>` `</div>` para representar esse container. E nele vamos utilizar a propriedade `display` como `flex` para assim criarmos um container que seja flexível. Conforme estrutura abaixo:

```
.container {  
  display: flex;  
}
```

Ou `display: inline-flex` (para criarmos um container que seja flexível a nível de linha). Conforme estrutura abaixo:

```
.container {  
  display: inline-flex;  
}
```

Propriedades do FlexBox

FLEX-DIRECTION:

A partir disso podemos utilizar a propriedade flex-direction para organizar os itens dentro do flex container. Temos 4 valores para essa propriedade:

- Row: organiza os itens em linha da esquerda para a direita:



A estrutura base do código que representa o **row** (o qual digitaremos dentro do arquivo **style.css**) será essa abaixo:

```
.container {  
  display: flex;  
  flex-direction: row;  
}
```

OBS.: Por padrão quando definimos o display como flex ele automaticamente entende que o flex-direction é row (então nesse caso específico não precisaria ser colocado o flex-direction: row.

Propriedades do FlexBox

- Row-Reverse: organiza os itens em linha da direita para a esquerda:

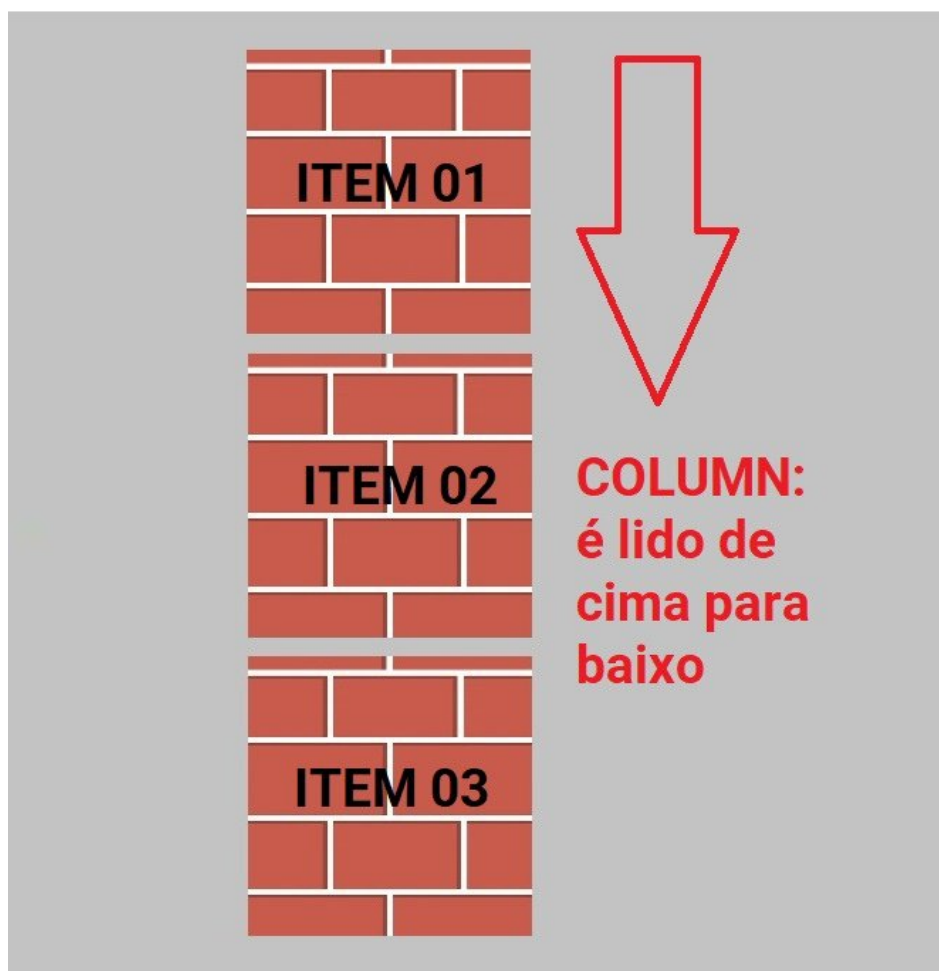


A estrutura base do código que representa o **row-reverse** (o qual digitaremos dentro do arquivo **style.css**) será essa abaixo:

```
.container {  
  display: flex;  
  flex-direction: row-reverse;  
}
```

Propriedades do FlexBox

- Column: organiza os itens em coluna de cima para baixo

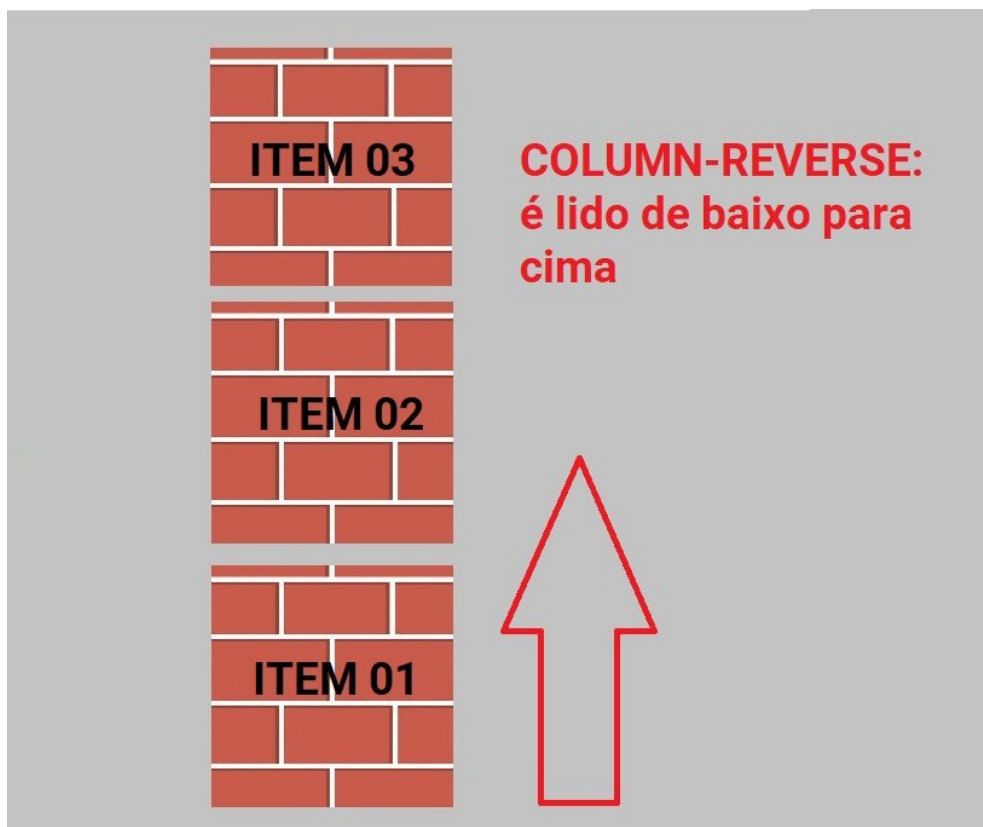


A estrutura base do código que representa o **column** (o qual digitaremos dentro do arquivo **style.css**) será essa abaixo:

```
.container {  
  display: flex;  
  flex-direction: column;  
}
```

Propriedades do FlexBox

- Column-Reverse: organiza os itens em coluna de baixo para cima:



A estrutura base do código que representa o **column-reverse** (o qual digitaremos dentro do arquivo **style.css**) será essa abaixo:

```
.container {  
  display: flex;  
  flex-direction: column-reverse;  
}
```

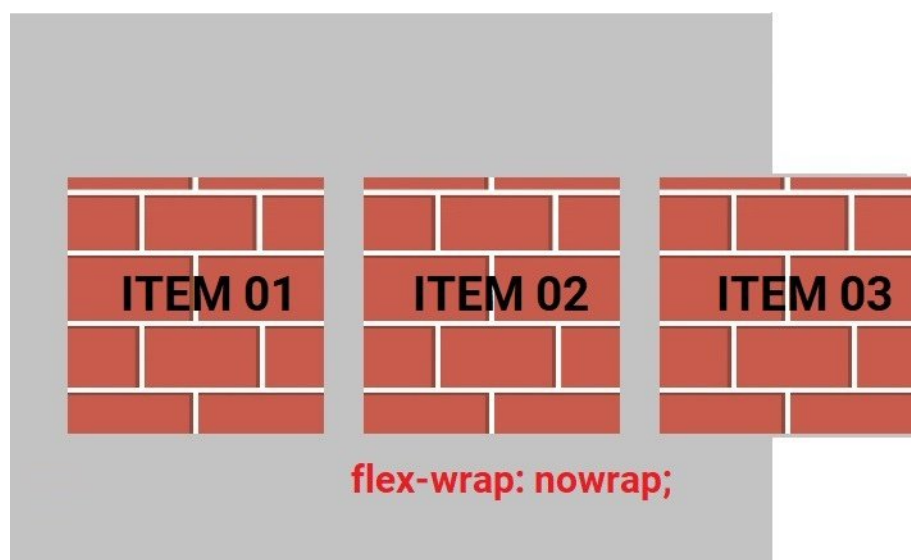

Propriedades do FlexBox

FLEX-WRAP:

No FlexBox a largura de um container ou de um item é ajustada automaticamente de acordo com o espaço disponível para eles. E para ajustar esse tipo de comportamento temos a propriedade flex-wrap que possui 3 valores:

- **nowrap** (esse valor já vem como padrão): e não é permitido a quebra de linha.

Notem que no exemplo abaixo temos a valor **nowrap** definido e o que aconteceu: O terceiro item do nosso revestimento(tijolinho) ficou "para fora" do nosso container parede, pois a largura dos (3 flex itens) são maiores do que o tamanho do nosso container parede e como o nowrap não permite a quebra de linha, ele "ajustou" dessa forma:

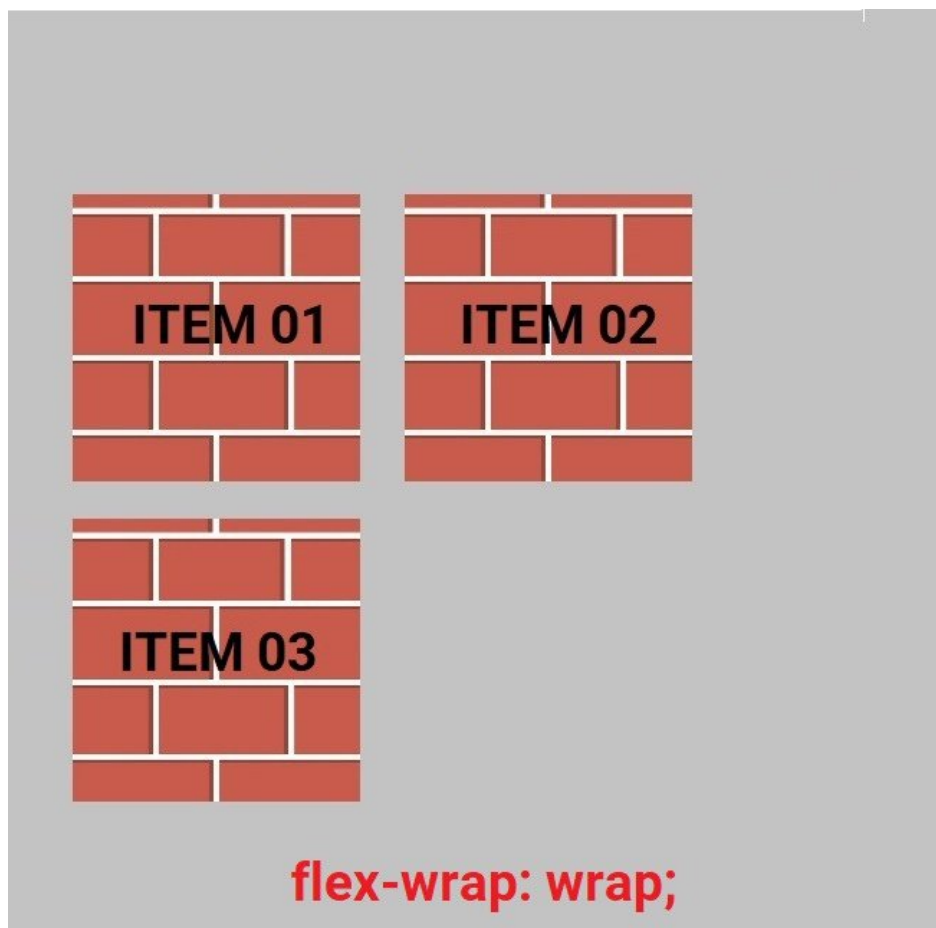


Propriedades do FlexBox

FLEX-WRAP:

- **wrap**: aqui a linha é quebrada e os itens que estão na parte mais a direita vão ser gentilmente convidados (leia-se: deslocados) para a linha abaixo.

Notem que no exemplo abaixo temos a valor wrap definido e o que aconteceu: O terceiro item do nosso revestimento(tijolinho) que está mais a direita do nosso container passou para linha de baixo pois usamos a propriedade wrap. E ele ficou ajustado assim:

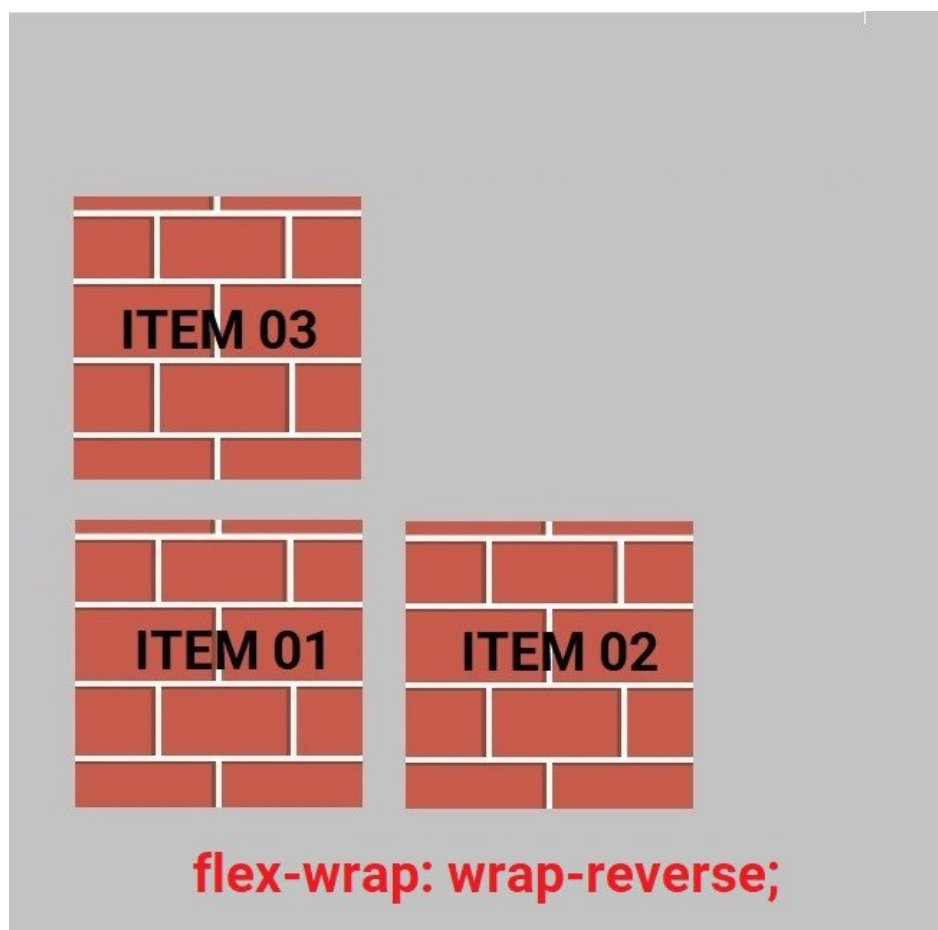


Propriedades do FlexBox

FLEX-WRAP:

- **wrap-reverse**: aqui a linha é quebrada e os itens que estão na parte mais a direita vão ser gentilmente convidados (leia-se: deslocados) para a linha acima

Notem que no exemplo abaixo temos a valor **wrap-reverse** definido e o que aconteceu: O terceiro item do nosso revestimento(tijolinho) que está mais a direita do nosso container passou para linha acima (de forma reversa) pois usamos a propriedade wrap-reverse. E ele ficou ajustado assim:



Propriedades do FlexBox

JUSTIFY-CONTENT:

– **justify-content**: vai definir o alinhamento dos itens em um container em relação ao eixo-principal. O justify-content ajuda a distribuir o espaço entre esses itens principalmente quando eles já atingiram o tamanho máximo.

A propriedade justify-content possui **5 valores** que poderão ser usados de acordo com a necessidade. Essas propriedades são:

justify-content: flex-start;

Aqui os itens são alinhados no início

justify-content: flex-end;

Aqui os itens são alinhados no final

justify-content: center;

Aqui os itens são alinhados ao centro

justify-content: space-between;

Aqui o primeiro item tem o seu deslocamento no início, enquanto o último item tem o seu deslocamento para o final e os itens restantes são distribuídos igualmente entre eles.

justify-content: space-around;

Aqui os itens são distribuídos igualmente entre eles. Com a diferença que o primeiro e o último item não vão ficar grudados nas bordas do container.

Vamos ver nas próximas páginas (as estruturas de cada uma das 5 propriedades do justify-content e um exemplo para melhor entendimento).



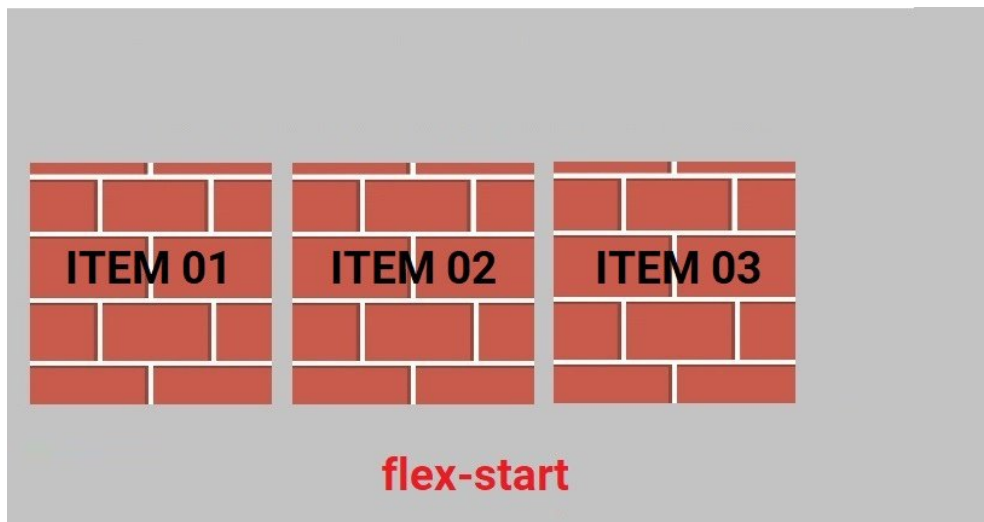
Propriedades do FlexBox

justify-content: flex-start;

Aqui os itens são alinhados no início

Estrutura básica:

```
. container {  
  display: flex;  
  justify-content: flex-start;
```



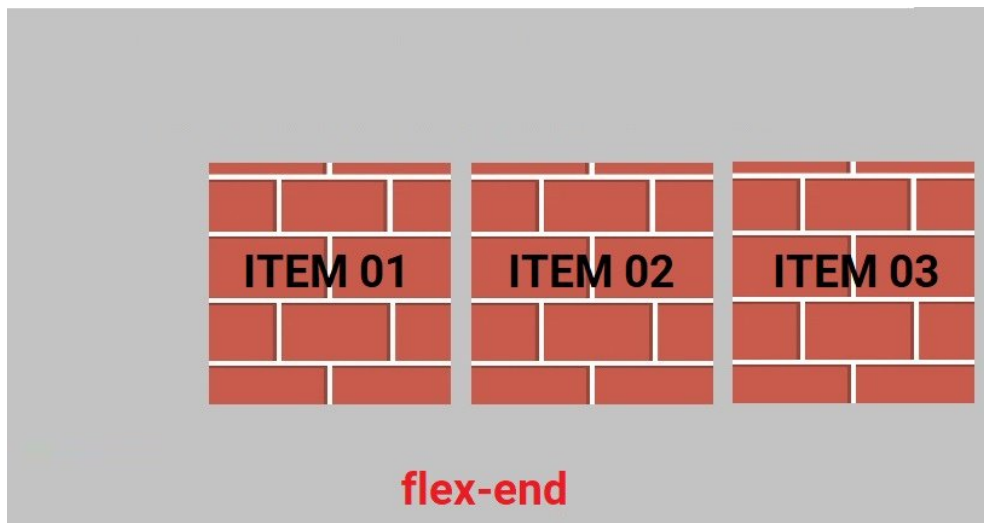
Propriedades do FlexBox

justify-content: flex-end;

Aqui os itens são alinhados no final

Estrutura básica:

```
. container {  
  display: flex;  
  justify-content: flex-end;  
}
```



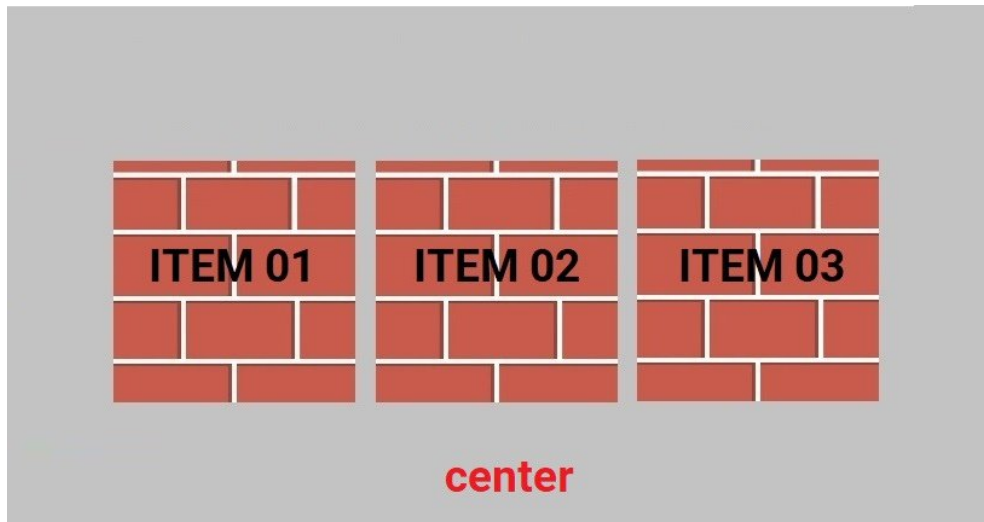
Propriedades do FlexBox

justify-content: center;

Aqui os itens são alinhados ao centro

Estrutura básica:

```
. container {  
  display: flex;  
  justify-content: center;  
}
```



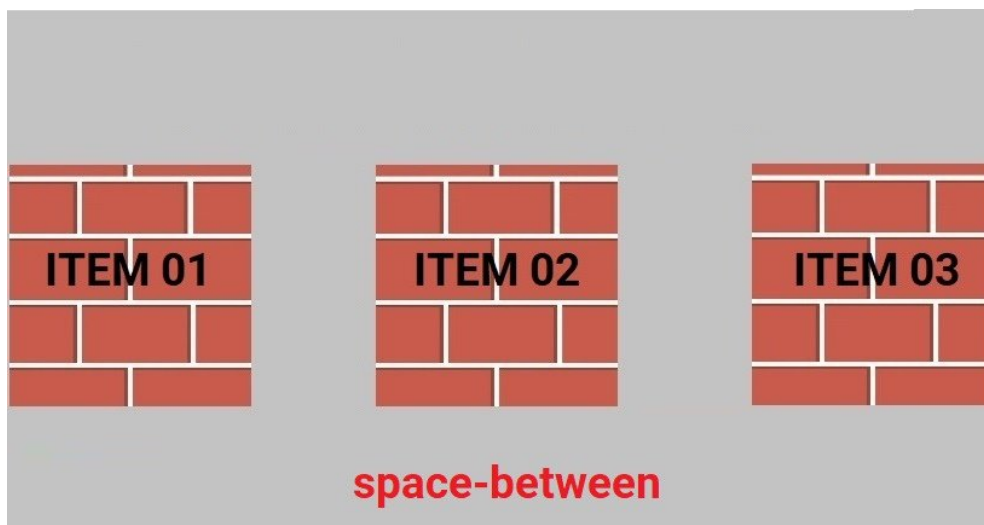
Propriedades do FlexBox

justify-content: space-between;

Aqui o primeiro item tem o seu deslocamento no início, enquanto o último item tem o seu deslocamento para o final e os itens restantes são distribuídos igualmente entre eles. Ops, quase me esqueci (o item mais a esquerda e o item mais a direita ficam grudados em suas respectivas bordas).

Estrutura básica:

```
. container {  
  display: flex;  
  justify-content: space-between;  
}
```



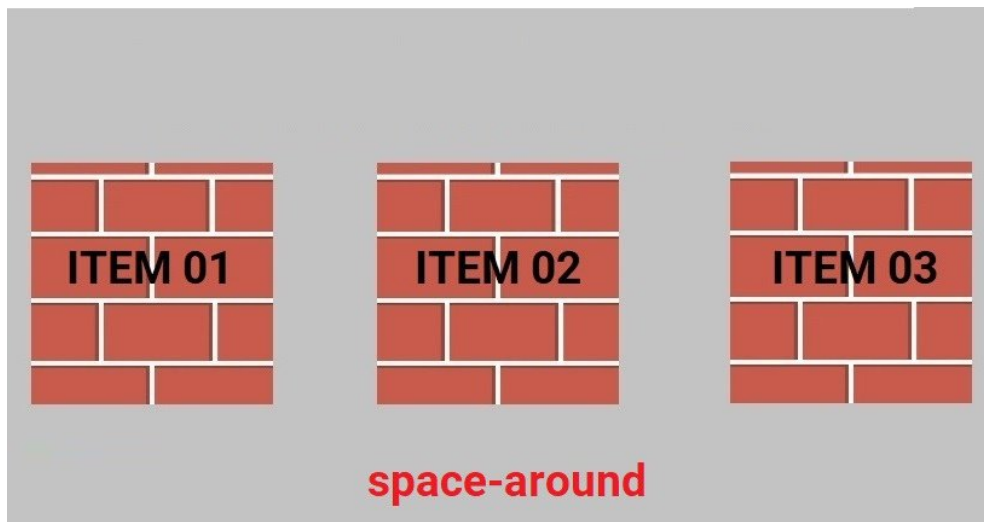
Propriedades do FlexBox

justify-content: space-around;

Aqui os itens são distribuídos igualmente entre eles. Com a diferença que o primeiro e o último item não vão ficar grudados nas bordas do container.

Estrutura básica:

```
. container {  
  display: flex;  
  justify-content: around;  
}
```



Exemplos do FlexBox

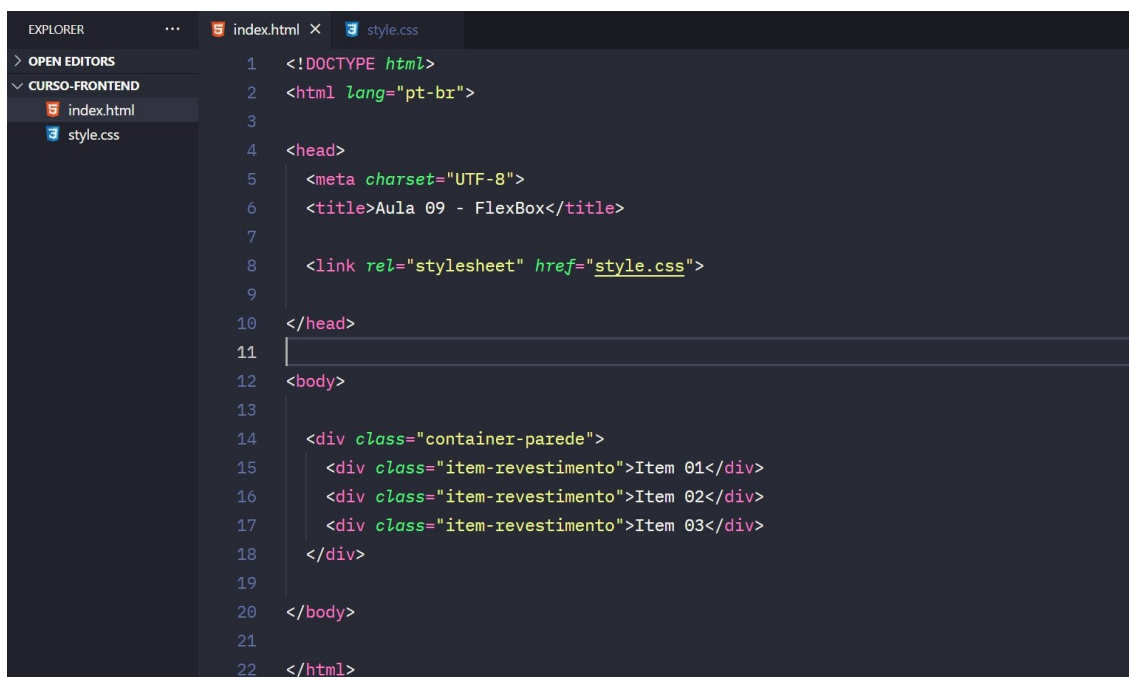
Vamos ver agora exemplos na prática dentro dos arquivos:

index.html (estrutura);

style.css (local onde vamos digitar as propriedades e os valores do FlexBox).

O código no index.html para exemplo terá essa estrutura, conforme imagem abaixo:

- 1 div (onde foi adicionada a class = container-parede)
- Outras 3 div's com a class= item-revestimento que estão dentro dessa class = container-parede.



```
1 <!DOCTYPE html>
2 <html lang="pt-br">
3
4 <head>
5   <meta charset="UTF-8">
6   <title>Aula 09 - FlexBox</title>
7
8   <link rel="stylesheet" href="style.css">
9
10 </head>
11
12 <body>
13
14   <div class="container-parede">
15     <div class="item-revestimento">Item 01</div>
16     <div class="item-revestimento">Item 02</div>
17     <div class="item-revestimento">Item 03</div>
18   </div>
19
20 </body>
21
22 </html>
```

Exemplos do FlexBox

Aqui a estrutura do código no arquivo style.css:

Na classe: container-parede (aqui no style.css representada pelo .container-parede), é composta por:

max-width: 500px

(definido uma largura máxima do nosso container-parede em 500px

margin: 0 auto;

(deixar centralizada quando renderizada no navegador)

border: 1px solid #black;

(definido uma borda de 1px, linha sólida na cor preta

background: grey;

(cor cinza para simular a cor de uma parede rebocada)

Na classe: item-revestimento (aqui no style.css representada pelo .item-revestimento) é composta por:

margin: 5px;

(para distanciar um item-revestimento dos outros)

padding: 5px;

(definimos um espaçamento interno)

background-color: orange;

(definimos a cor de fundo em laranja para simular o revestimento tijolinho)

text-align: center;

(texto alinhado no centro)

font-size: 16px;

(tamanho da fonte definido em 16px)

color: white;

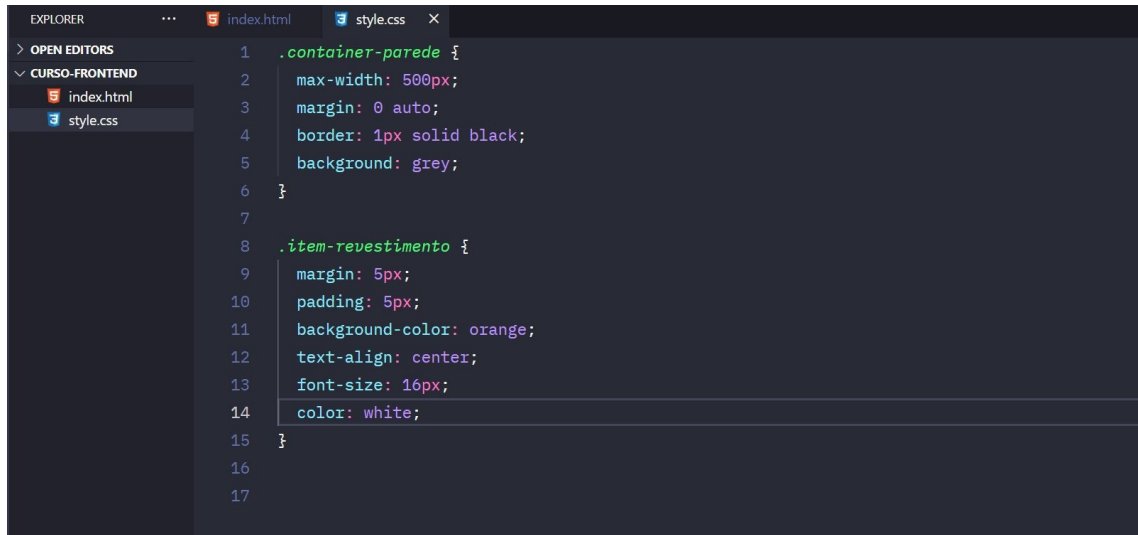
(cor da fonte na cor branca)



/igor-rebolla

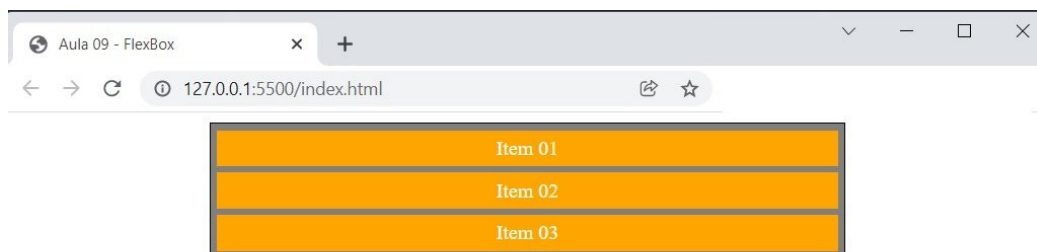
Exemplos do FlexBox

O código no style.css inicialmente terá essa estrutura, conforme imagem abaixo:



```
1 .container-parede {  
2   max-width: 500px;  
3   margin: 0 auto;  
4   border: 1px solid black;  
5   background: grey;  
6 }  
7  
8 .item-revestimento {  
9   margin: 5px;  
10  padding: 5px;  
11  background-color: orange;  
12  text-align: center;  
13  font-size: 16px;  
14  color: white;  
15 }  
16  
17
```

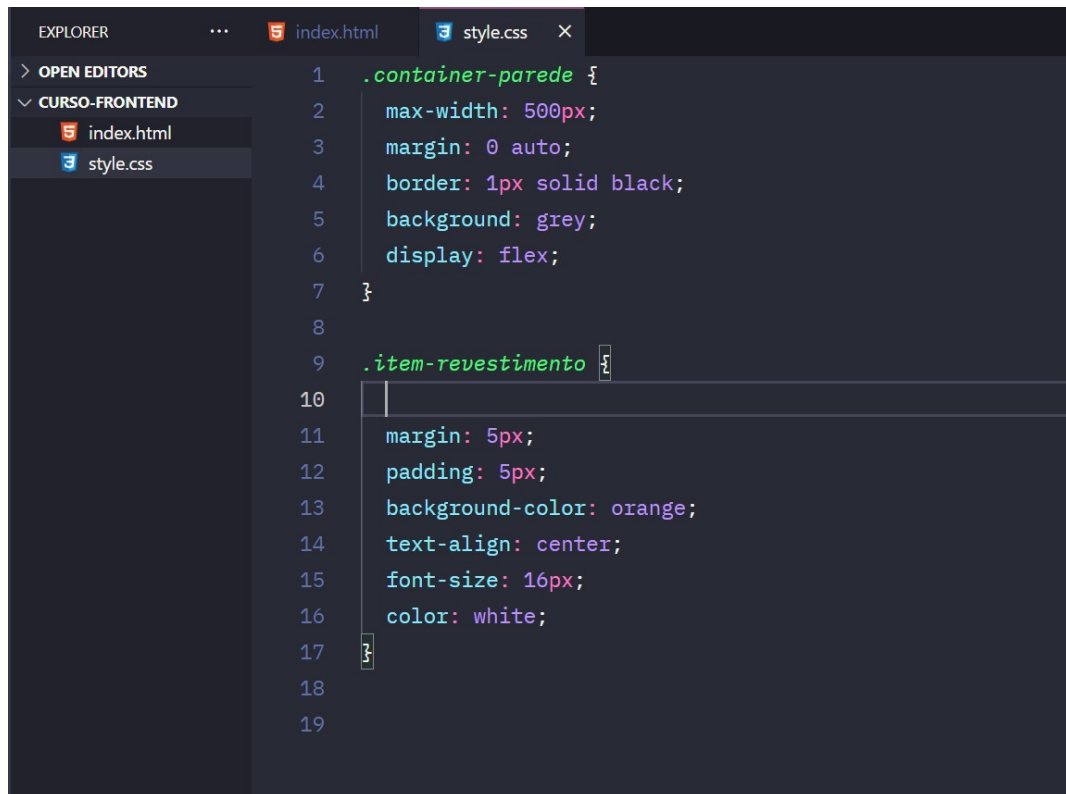
Na imagem abaixo podemos ver o resultado desse código renderizado no navegador



Notem que não colocamos nenhuma propriedade do FlexBox vista nessa aula até o momento. Vamos fazer isso a partir da próxima página.

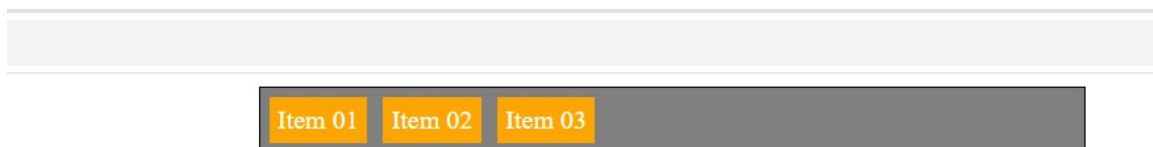
Exemplos do FlexBox

Para começarmos a usar o **FlexBox**, precisamos definir antes de tudo o nosso **container-parede** como **display:flex**;



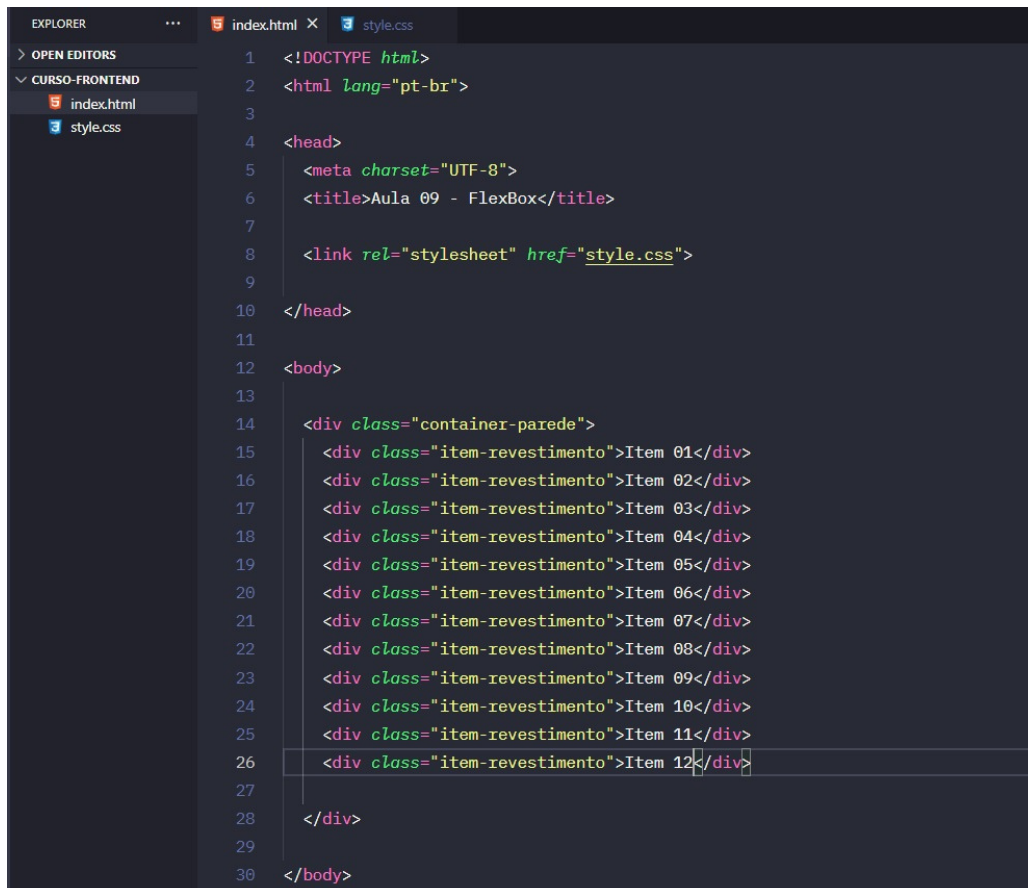
```
1 .container-parede {
2   max-width: 500px;
3   margin: 0 auto;
4   border: 1px solid black;
5   background: grey;
6   display: flex;
7 }
8
9 .item-revestimento {
10
11   margin: 5px;
12   padding: 5px;
13   background-color: orange;
14   text-align: center;
15   font-size: 16px;
16   color: white;
17 }
18
19
```

Abaixo vamos ver o código acima renderizado no navegador depois que colocamos o `display:flex`; (os itens que antes estavam em bloco por padrão agora estão em linha):



Exemplos do FlexBox

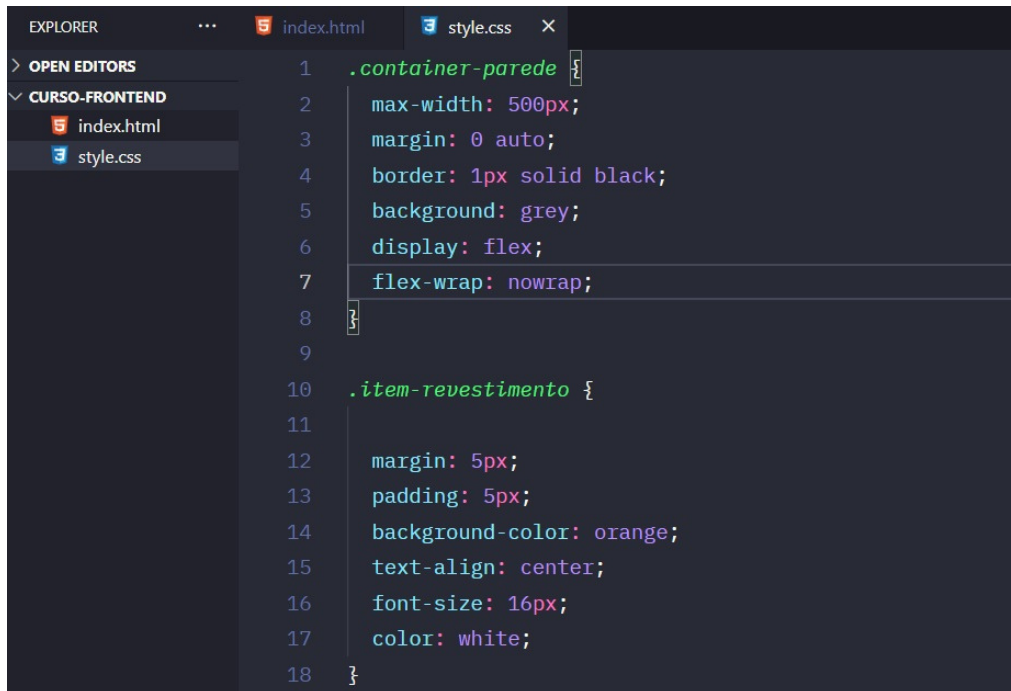
Para vermos na prática o funcionamento do nowrap (vamos acrescentar mais 9 div's no nosso arquivo index.html)



```
1 <!DOCTYPE html>
2 <html lang="pt-br">
3
4 <head>
5   <meta charset="UTF-8">
6   <title>Aula 09 - FlexBox</title>
7
8   <link rel="stylesheet" href="style.css">
9
10 </head>
11
12 <body>
13
14   <div class="container-parede">
15     <div class="item-revestimento">Item 01</div>
16     <div class="item-revestimento">Item 02</div>
17     <div class="item-revestimento">Item 03</div>
18     <div class="item-revestimento">Item 04</div>
19     <div class="item-revestimento">Item 05</div>
20     <div class="item-revestimento">Item 06</div>
21     <div class="item-revestimento">Item 07</div>
22     <div class="item-revestimento">Item 08</div>
23     <div class="item-revestimento">Item 09</div>
24     <div class="item-revestimento">Item 10</div>
25     <div class="item-revestimento">Item 11</div>
26     <div class="item-revestimento">Item 12</div>
27
28   </div>
29
30 </body>
```

Agora vamos acrescentar no arquivo style.css a propriedade flex-wrap: nowrap; (conforme imagem da próxima página).

Exemplos do FlexBox



```
EXPLORER  ...  index.html  style.css  X
> OPEN EDITORS
CURSO-FRONTEND
  index.html
  style.css

1  .container-parede {
2      max-width: 500px;
3      margin: 0 auto;
4      border: 1px solid black;
5      background: grey;
6      display: flex;
7      flex-wrap: nowrap;
8  }
9
10 .item-revestimento {
11     margin: 5px;
12     padding: 5px;
13     background-color: orange;
14     text-align: center;
15     font-size: 16px;
16     color: white;
17 }
18 }
```

Vamos ver o resultado do código acima renderizado no navegador (conforme imagem abaixo).



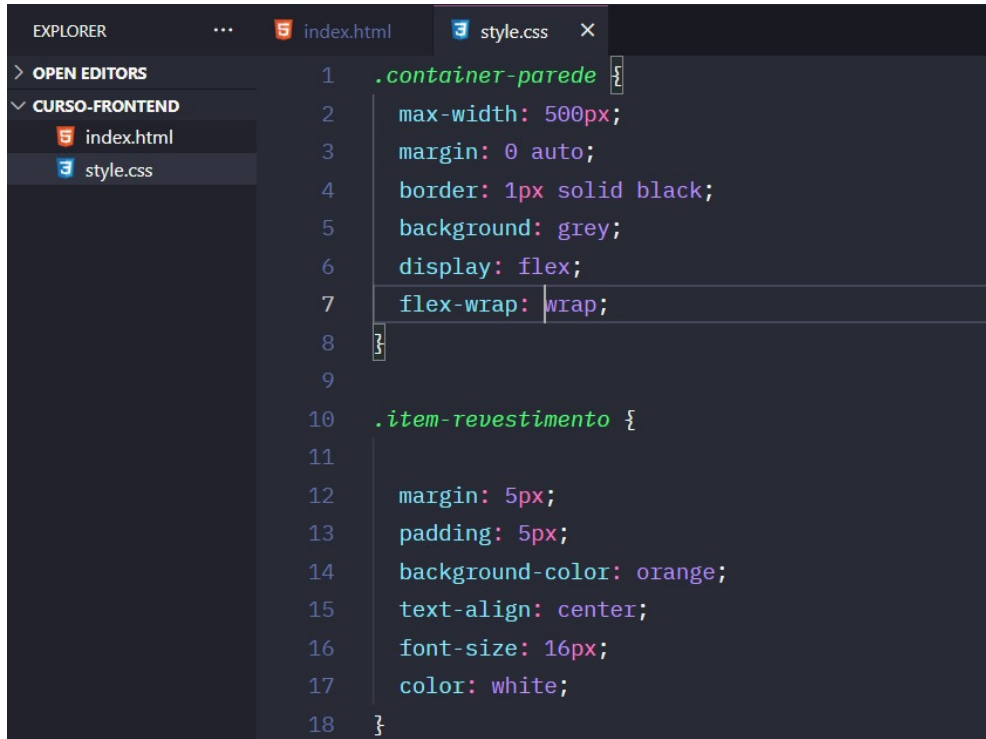
Qual o motivo do item 11 e 12 terem saído do nosso container-parede? Por causa do nowrap (esse valor não permite quebra de linha) e ele continua como "senão" houvesse o amanhã sempre em frente caso tivéssemos acrescentado mais 3 div's por exemplo. Para resolver esse "problema" podemos usar wrap (o qual veremos na próxima página).

Partiu ver o wrap...

Exemplos do FlexBox

Agora vamos acrescentar no arquivo style.css a propriedade flex-wrap: wrap; (conforme imagem abaixo).

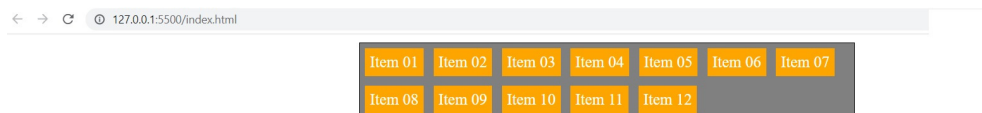
Ah e vamos manter as 12 div's no arquivo index.html.



```
EXPLORER  ...  index.html  style.css x
> OPEN EDITORS
CURSO-FRONTEND
  index.html
  style.css

1  .container-parede {
2      max-width: 500px;
3      margin: 0 auto;
4      border: 1px solid black;
5      background: grey;
6      display: flex;
7      flex-wrap: wrap;
8  }
9
10 .item-revestimento {
11
12     margin: 5px;
13     padding: 5px;
14     background-color: orange;
15     text-align: center;
16     font-size: 16px;
17     color: white;
18 }
```

Vamos ver o resultado do código acima renderizado no navegador (conforme imagem abaixo).



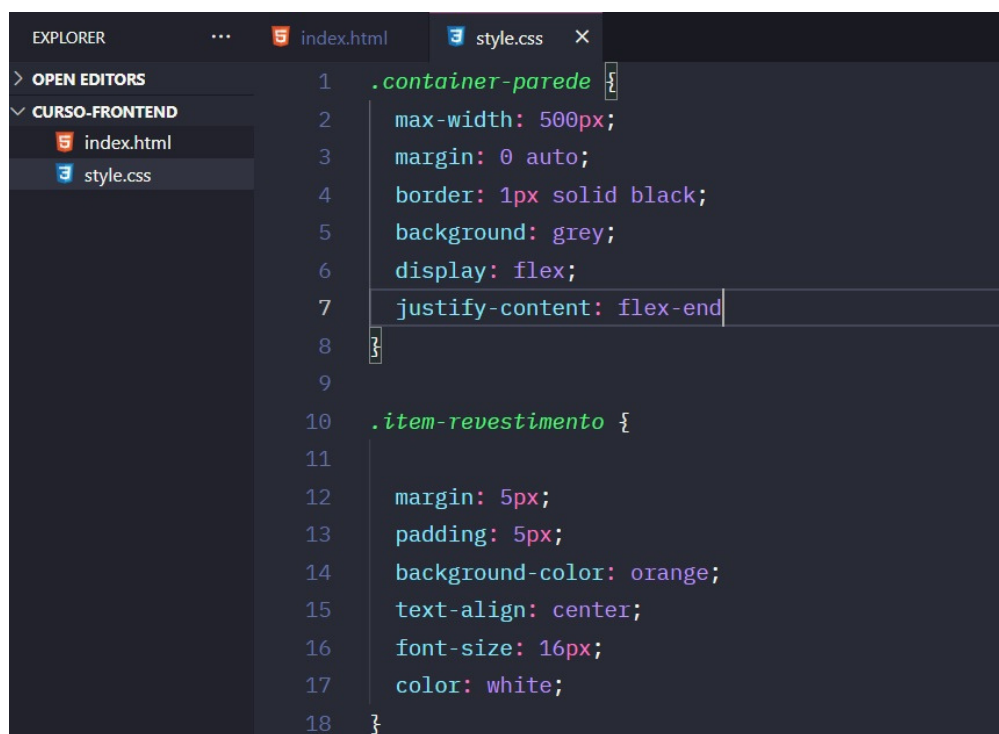
O item 07 foi o último item que cabia no container sendo assim o item 08 foi gentilmente convidado a descer (pois ele não cabia mais na linha de cima) e automaticamente levou o item 09 até o 12 com ele.

Exemplos do FlexBox

Agora vamos ver um exemplo de uma das 5 propriedades do justify-content que vimos nessa aula. Bora ver o justify-content: flex-end;

Ah e vamos voltar com as nossas 3 div's no arquivo index.html.

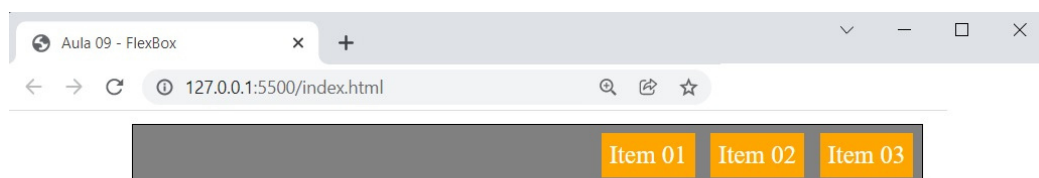
Depois que colocarmos o justify-content: flex-end; (dentro do nosso arquivo style.css) ele vai ficar conforme imagem abaixo:



```
EXPLORER  ...  index.html  style.css  X
> OPEN EDITORS
CURSO-FRONTEND
  index.html
  style.css

1  .container-parede {
2      max-width: 500px;
3      margin: 0 auto;
4      border: 1px solid black;
5      background: grey;
6      display: flex;
7      justify-content: flex-end;
8  }
9
10 .item-revestimento {
11
12     margin: 5px;
13     padding: 5px;
14     background-color: orange;
15     text-align: center;
16     font-size: 16px;
17     color: white;
18 }
```

Vamos ver o resultado do código acima renderizado no navegador (conforme imagem abaixo).



Links das 8 aulas anteriores

Para acessar cada aula, basta clicar na aula de interesse abaixo que será direcionado diretamente para o post com a respectiva aula no LinkedIn:

[Aula 01](#)

[Aula 02](#)

[Aula 03](#)

[Aula 04](#)

[Aula 05](#)

[Aula 06](#)

[Aula 07](#)

[Aula 08](#)



Sugestões de prática para essa aula:

1. Abrir o Microsoft Visual Studio Code, depois a pasta Curso-FrontEnd e os arquivos: index.html e style.css;
2. Nas páginas 18 até 25 dessa aula eu fiz alguns exemplos das propriedade explicadas nas páginas anteriores. Minha sugestão aqui é: Fazer os exemplos que ficaram faltando:

- flex-wrap: wrap-reverse;
 - justify-content: flex-start;
 - justify-content: center;
 - justify-content: space-between;
- (Obs para o space-between: Na explicação desse valor na página 16 dessa aula eu citei que o space-between tem o item mais a esquerda grudado na sua respectiva borda e o item mais a direita a mesma (isso é o padrão da propriedade) nada impede de colocar uma margin de 5px por exemplo e dar respectivos espaçamentos.
- justify-content: space-around;
 - flex-direction: row;
 - flex-direction: row-reverse;
 - flex-direction: column;
 - flex-direction: column-reverse;

3. Ah e bora fazer os exemplos mostrados nessa aula também.

O que vamos aprender na aula 10:

Na décima e próxima aula, continuaremos nossos estudos sobre o Segundo Pilar do Curso - O CSS 3

Feedbacks dessa nona aula são bem-vindos.

Nos vemos na décima aula que será disponibilizada no dia 03/02/2022 às 8h30 (Horário de Brasília) - Dentro do meu perfil no LinkedIn.

