



FLEXBOX



Flexbox

Michele Queiroz Ambrosio

Desenvolvedora front-end

@programi_

Sobre mim

(2012-2013)

Curso técnico em informática na ETEC;

(2014-2017)

Faculdade de Ciência da Computação na UNISO;

(2017-atualmente)

Desenvolvedora Front-end na Eduzz.

Sobre mim

Como você pode me encontrar
nas redes sociais?

Instagram: @programi_

Twitch: twitch.tv/michele_ambrosio

LinkedIn: Michele Ambrosio

Github: @micheleambrosio



Objetivo geral

O objetivo geral deste curso é ensinar os principais conceitos, aplicações e propriedades que envolvem o flexbox.

Pré-requisitos

É importante que você já tenha uma base de HTML e assistido ao **Módulo 1 de CSS**, juntamente com o curso de **Posicionamentos e Displays com CSS** para acompanhar os exemplos e conseguir estilizar suas páginas.

Ferramentas utilizadas

- VSCode;
- Plugins para VSCode: Live Server e Emmet;
- Google Chrome.

Percurso

Etapa 1

Entendendo os conceitos do flexbox

Etapa 2

Os eixos do flexbox

Etapa 3

Definindo a direção do flexbox com a propriedade flex-direction

Etapa 4

Quebra de linhas e colunas com a propriedade flex-wrap

Percurso

Etapa 5

Juntando a propriedade flex-direction e a flex-wrap em uma só propriedade abreviada (flex-flow)

Etapa 6

Alinhando os elementos no eixo principal com justify-content

Etapa 7

Alinhamento de elementos no eixo transversal com align-items

Etapa 8

Alinhando as linhas no container com align-content

Percurso

Etapa 9

Controlando os espaços entre os itens com gap

Etapa 10

Ordenando os itens com a propriedade order

Etapa 11

Propriedade flex-grow

Etapa 12

Propriedade flex-shrink

Percurso

Etapa 13

Propriedade flex-basis

Etapa 14

Shorthand flex

Etapa 15

Alinhando itens individualmente com align-self

Etapa 16

Materiais de Apoio

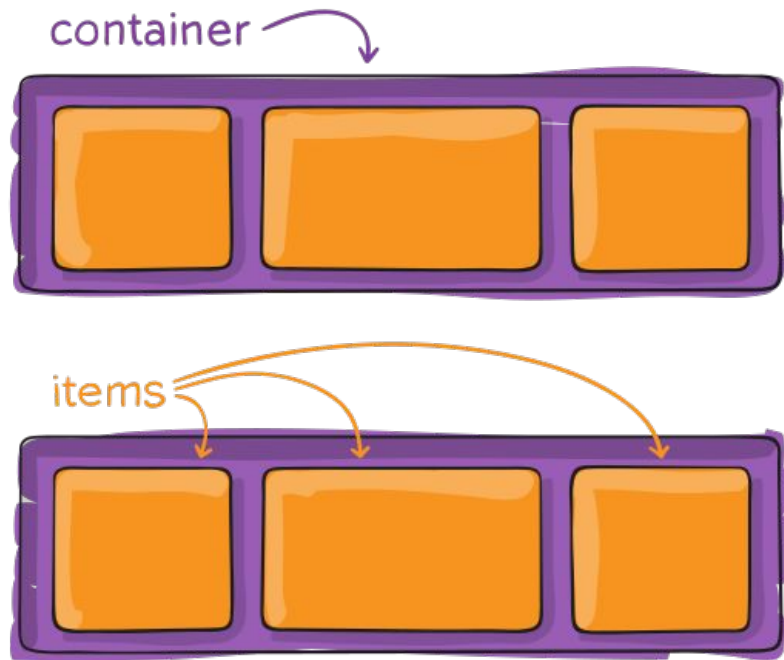
Etapa 1

Conceitos do flexbox

Conceitos do flexbox

Permite posicionar os elementos dentro de outro elemento (*container*);

Os elementos dentro do *container* são conhecidos como itens.



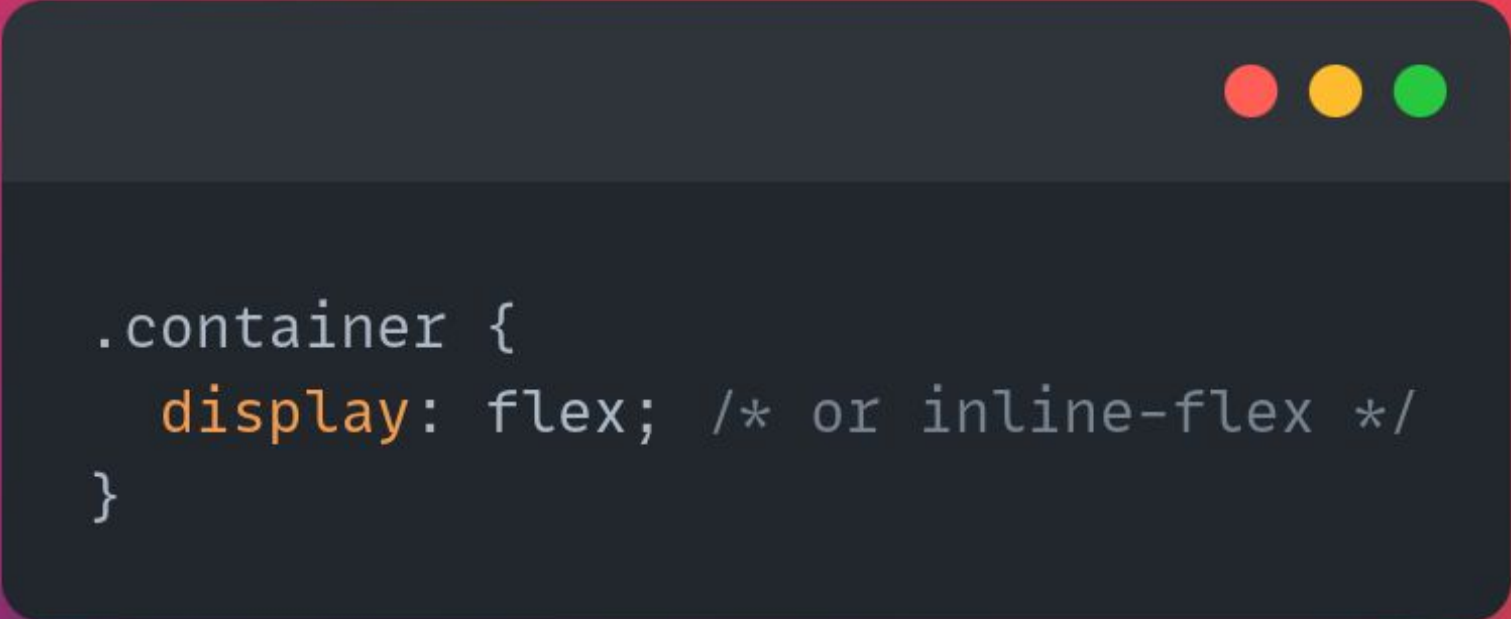
Conceitos do flexbox

O *flexbox* (*Flexible Box Module*) oferece uma maneira mais eficiente de organizar, alinhar e distribuir o espaço entre os itens do *container*, mesmo que o tamanho do elemento seja dinâmico.

Conceitos do flexbox

Permite trazer para o *container* a possibilidade de alterar a largura/altura; ordem dos itens e aproveitar o espaço disponível dentro do elemento da melhor forma.



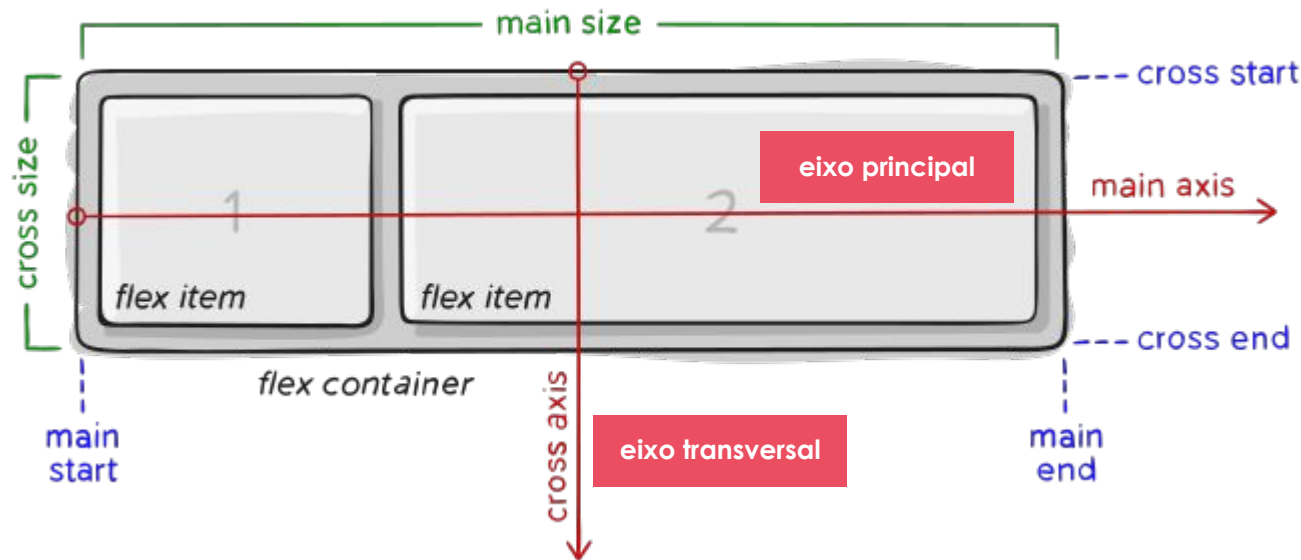


```
.container {  
  display: flex; /* or inline-flex */  
}
```


Etapa 2

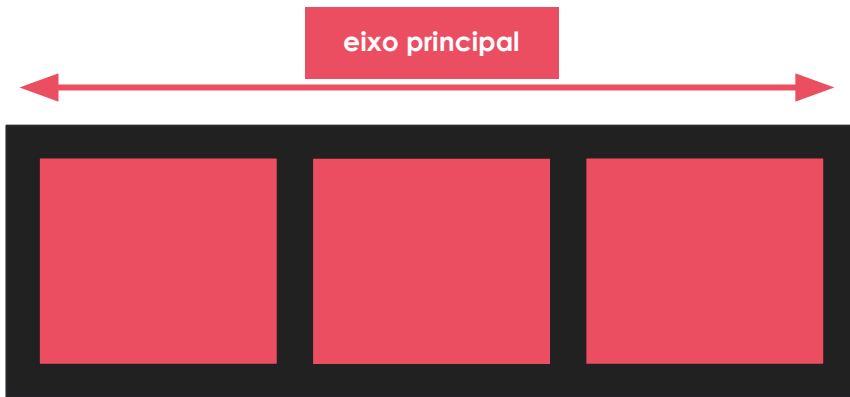
Os eixos do flexbox

Os eixos do flexbox



Eixo principal (main axis)

Define a direção em que o nosso container vai organizar os nossos itens: horizontalmente (em linha), ou verticalmente (em coluna).



Eixo transversal (cross axis)

O eixo transversal é perpendicular (atravessa) ao eixo principal.



Etapa 3

**Definindo a direção do
flexbox com a propriedade
flex-direction**

Propriedade flex-direction

flex-direction: row



flex-direction: row-reverse



flex-direction: column

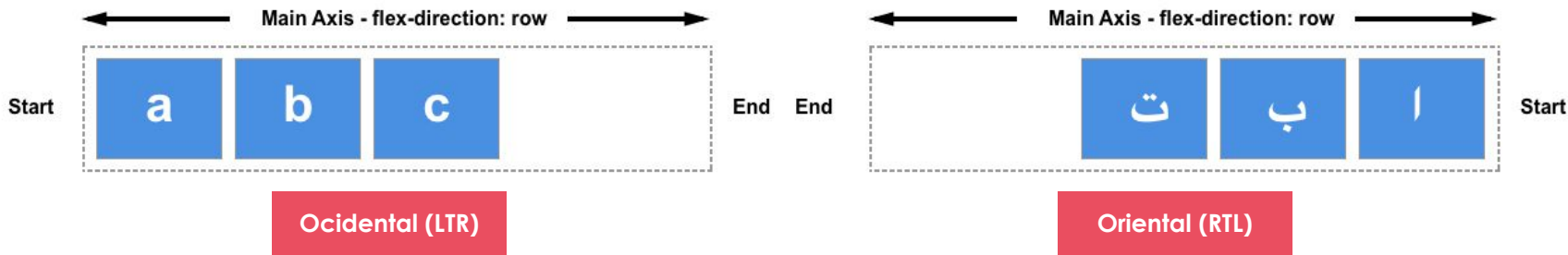


flex-direction:
column-reverse



Estilo da escrita & flexbox

A propriedade **flex-direction** vai levar em consideração o estilo de escrita no qual estamos trabalhando no projeto para definir se, **flex-direction: row**, por exemplo, deve ser da esquerda para direita, ou da direita para a esquerda.



Etapa 6

**Alinhando os elementos no
eixo principal com
justify-content**

Valores de justify-content

flex-start



flex-end



center



space-between



space-around

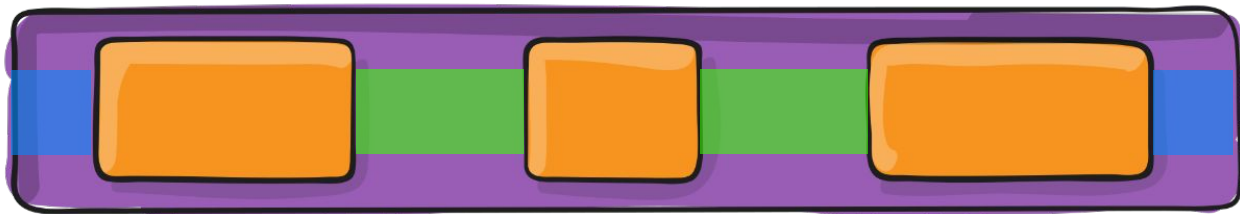


space-evenly

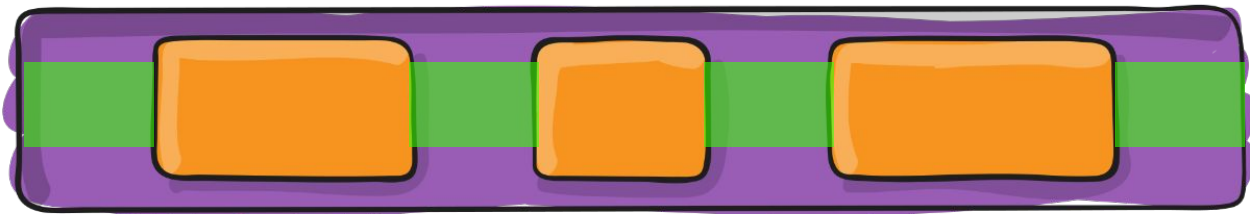


space-around x space-evenly

space-around



space-evenly

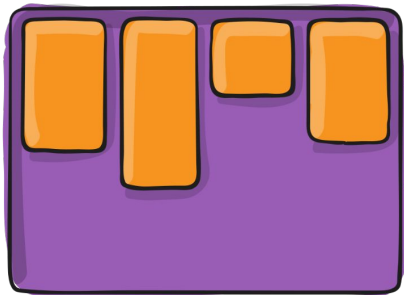


Etapa 7

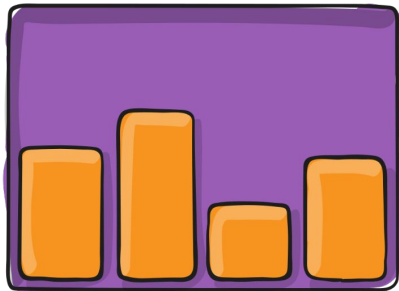
**Alinhando os elementos no
eixo transversal com
align-items**

Valores de align-items

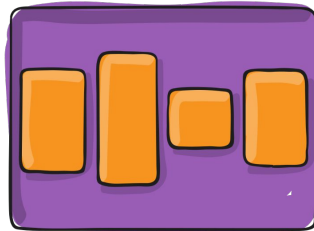
flex-start



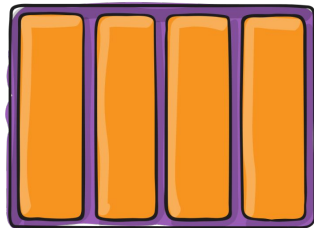
flex-end



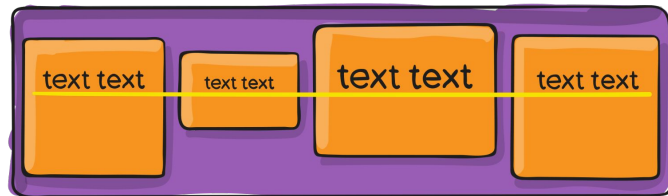
center



stretch



baseline

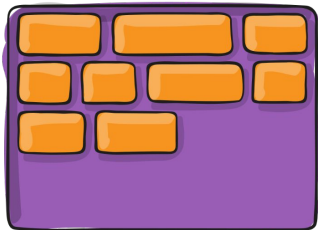


Etapa 8

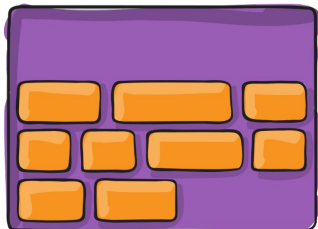
**Alinhando as linhas no
container com align-content**

Valores de align-content

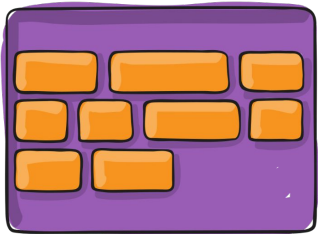
flex-start



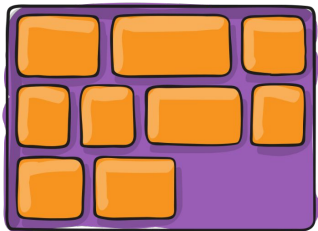
flex-end



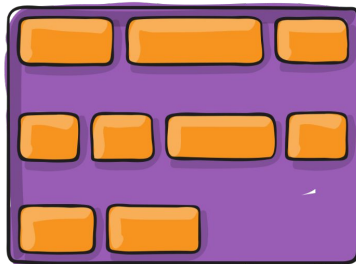
center



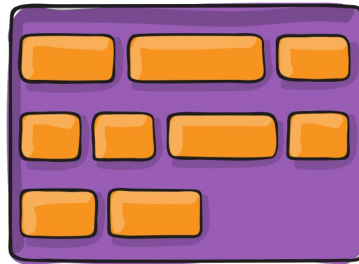
stretch



space-between



space-around

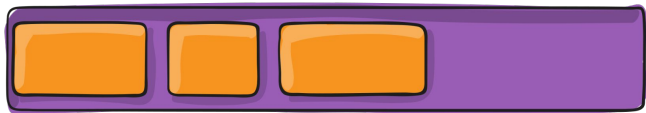


Etapa 9

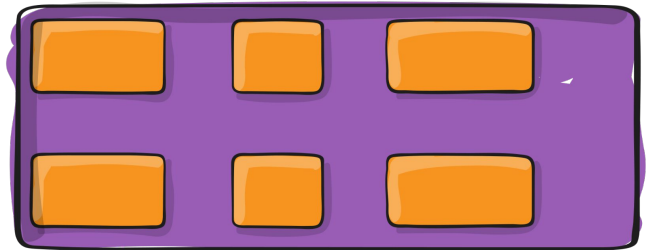
Controlando os espaços
entre itens com gap

Valores de gap

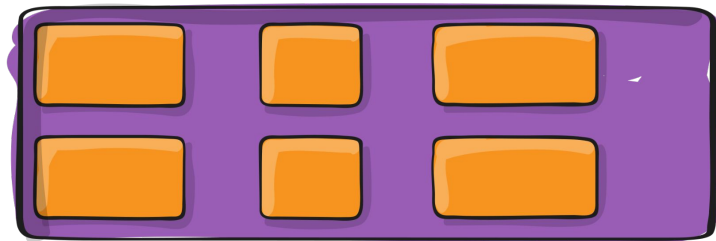
gap: 10px



gap: 30px



gap: 10px 30px





Etapa 10

Ordenando os itens do *container*

Ordenando os itens no container

Podemos ordenar os *flex-items* através da propriedade **order**.
Por padrão, os itens são exibidos conforme colocamos no HTML, mas, podemos ordená-lo individualmente.

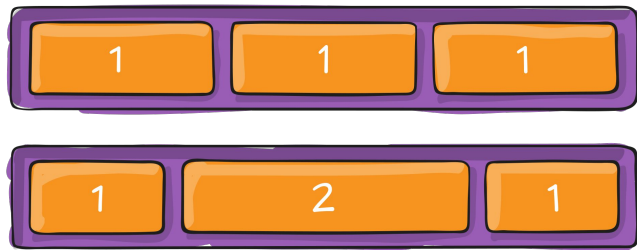


Etapa 11

Propriedade flex-grow

Propriedade flex-grow

A propriedade **flex-grow** fará com que esse espaço extra seja usado por esses itens, dando a eles a capacidade para o item de crescer, ao longo do eixo principal, caso for necessário.



Atenção: Números negativos não serão aceitos

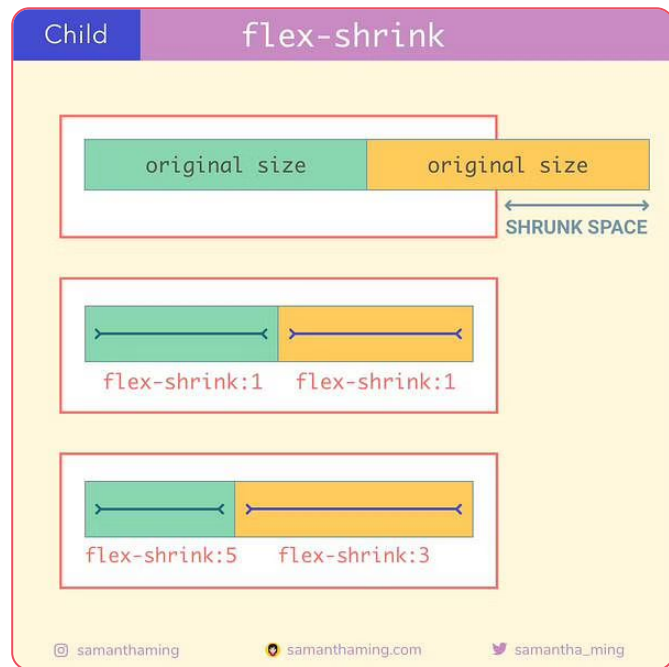
Etapa 12

Propriedade flex-shrink

Propriedade flex-shrink

A propriedade **flex-shrink** vai controlar o quanto esse item vai diminuir, caso seja necessário.

Se o *container* não possuir espaço suficiente para acomodar os elementos, eles irão diminuir para caber.



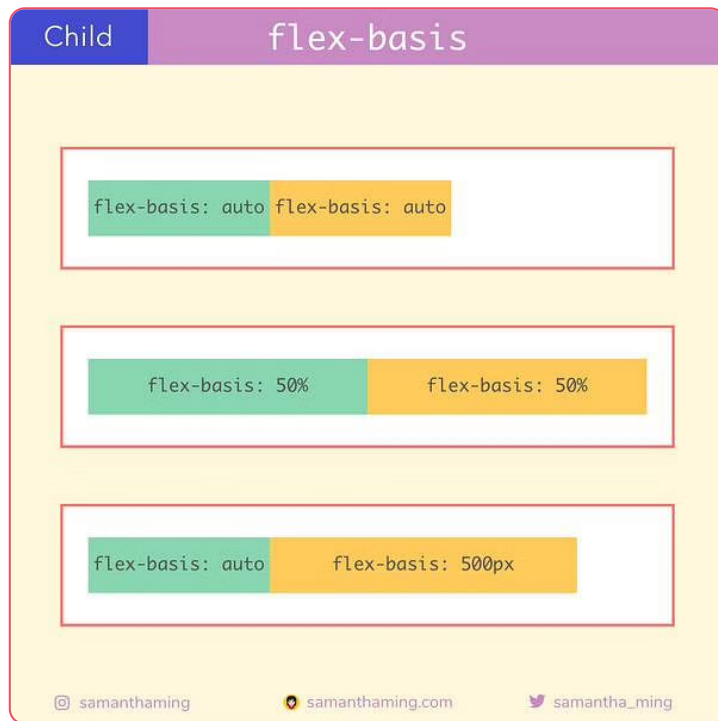
Etapa 13

Propriedade flex-basis

Propriedade flex-basis

A propriedade **flex-basis** define qual o tamanho inicial dos itens do nosso flexbox, antes que o espaço que de sobra seja redistribuído no *container*, de acordo com sua dimensão.

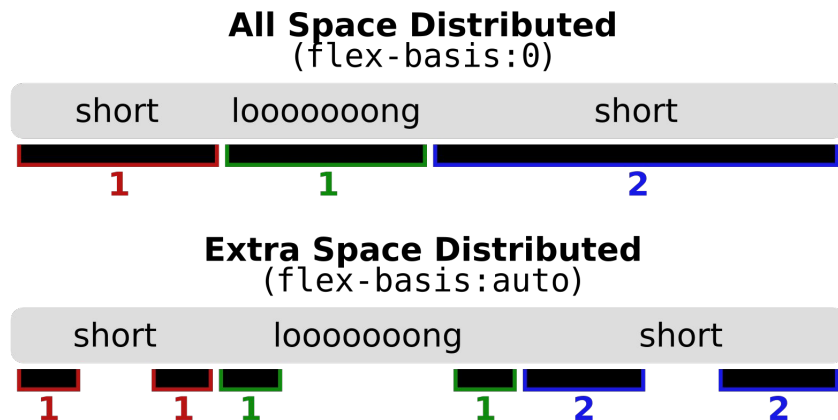
Pode ter um valor absoluto ou relativo para essa propriedade.



Flex-basis: distribuição do espaço

Caso seja aplicado

flex-basis: 0, o espaço extra em volta do conteúdo não será considerado. Se definirmos como **flex-basis: auto**, o espaço extra será distribuído com base no que definimos na propriedade **flex-grow**.



flex-basis x width/height

Se usarmos a propriedade **flex-basis** no item, ele irá sobrescrever o valor do **width/height** do elemento.

Porém, vai respeitar o **max-width/max-height** definido para o elemento, assim como o **min-width/min-height**.

flex-basis vs widths

<pre>.child { ✓ flex-basis: 500px width: 100px }</pre>	<p>flex-basis wins</p> <div>500px</div>
<pre>.child { ✓ min-width: 100px flex-basis: 500px }</pre>	<p>min-width wins</p> <div>100px</div>
<pre>.child { ✓ max-width: 700px flex-basis: 500px }</pre>	<p>max-width wins</p> <div>700px</div>

© samanthaming samanthaming.com samantha_ming

Etapa 14

Propriedade flex (*shorthand*)

Propriedade flex (*shorthand*)

Normalmente, as propriedades **flex-grow**, **flex-shrink** e **flex-basis** não são usadas de forma individual, mas sim através da propriedade abreviada **flex**.

Os valores de **flex-grow**, **flex-shrink** e **flex-basis**, são definidos, respectivamente, através da propriedade **flex**.

Child

flex

`flex: flex-grow flex-shrink flex-basis`

`flex: flex-grow`
ONE VALUE, UNITLESS

`flex: flex-basis`
ONE VALUE: UNIT

`flex: flex-grow flex-basis`
TWO VALUES: UNITLESS & UNIT

`flex: flex-grow flex-shrink`
TWO VALUES: UNITLESS & UNITLESS

 samanthaming

 samanthaming.com

 samantha_ming

Etapa 15

**Alinhando os itens
individualmente com
align-self**

Alinhamento com align-self

A propriedade **align-self** permite que nós possamos sobrescrever o alinhamento padrão aplicado em todos os itens de forma geral que foi definido pela propriedade **align-items**.

