Apresentação sobre Flexbox

Heder Dorneles Soares

Instituto Federal de São Paulo

19 de setembro de 2023

Introdução ao Flexbox

O módulo Flexbox Layout (ou "Caixa Flexível"), uma Recomendação Candidata da W3C desde outubro de 2017, tem como objetivo fornecer uma maneira mais eficiente de organizar, alinhar e distribuir espaço entre itens em um contêiner, mesmo quando o tamanho deles é desconhecido e/ou dinâmico (daí a palavra "flexível").

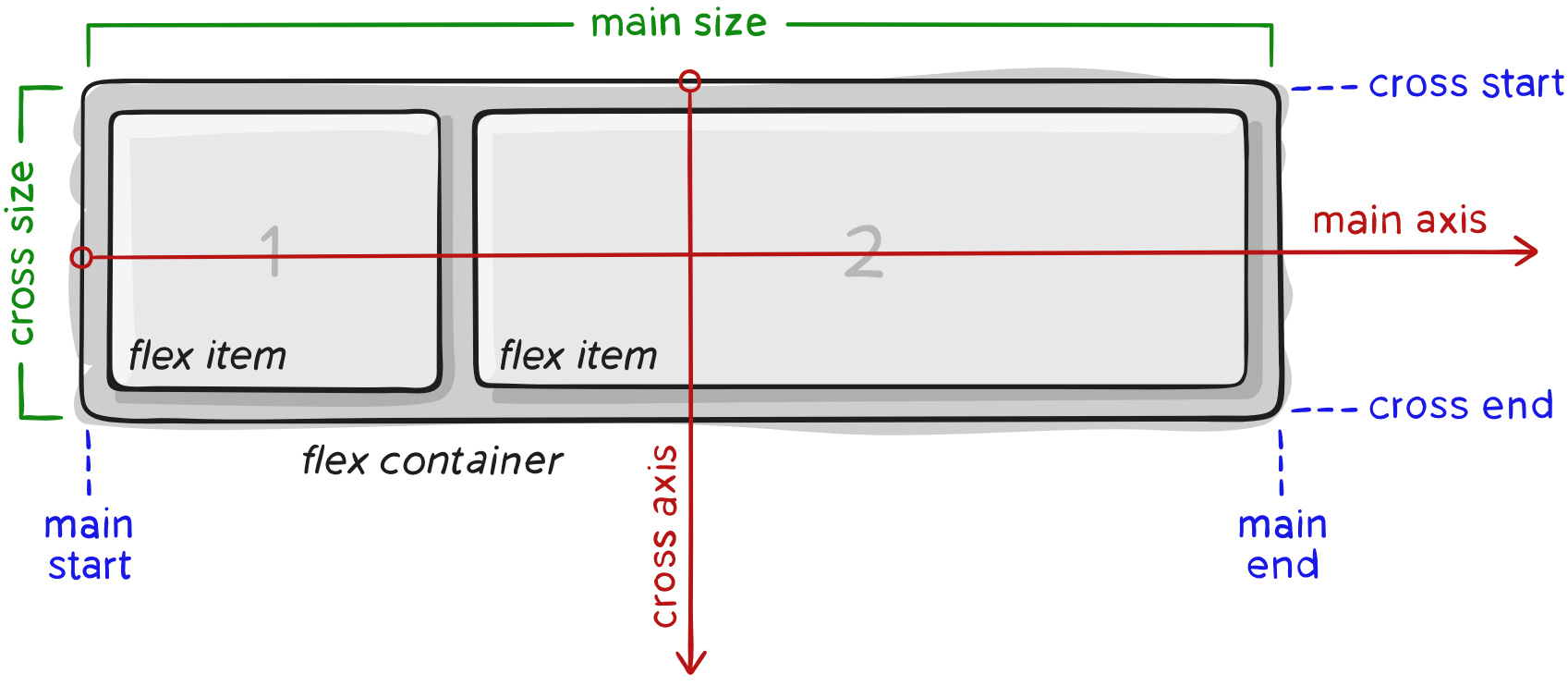
A Ideia Principal por Trás do Flexbox

A ideia principal por trás do layout flexível (Flexbox) é dar ao contêiner a capacidade de alterar a largura/altura (e ordem) de seus itens para preencher da melhor forma o espaço disponível (principalmente para se adaptar a todos os tipos de dispositivos de exibição e tamanhos de tela). Um contêiner flexível expande os itens para preencher o espaço livre disponível ou os reduz para evitar transbordamento.

Conceitos Básicos e Terminologia

Como o flexbox é um módulo completo e não uma única propriedade, ele envolve muitas coisas, incluindo todo o conjunto de suas propriedades. Algumas delas são destinadas a serem definidas no contêiner (elemento pai, conhecido como "contêiner flex") enquanto outras são destinadas a serem definidas nos filhos (chamados de "itens flexíveis").
Se o layout "regular"é baseado nas direções de fluxo de bloco e inline, o layout flex é baseado em "direções de fluxo flex". Por favor, dê uma olhada nesta figura da especificação, explicando a ideia principal por trás do layout flex.

Especificação



Terminologia do Flexbox

- **main axis** - O eixo principal de um contêiner flex é o eixo primário ao longo do qual os itens flexíveis são dispostos. Cuidado, ele não é necessariamente horizontal; depende da propriedade flex-direction (ver abaixo).
- **main-start — main-end** - Os itens flexíveis são posicionados dentro do contêiner começando a partir do início principal e indo até o fim principal.
- **main size** - A largura ou altura de um item flexível, o que estiver na dimensão principal, é o tamanho principal do item. A propriedade do tamanho principal do item flexível é a propriedade 'width' ou 'height', o que estiver na dimensão principal.

Terminologia do Flexbox (Continuação)

- **cross axis** - O eixo perpendicular ao eixo principal é chamado de eixo cruzado. Sua direção depende da direção do eixo principal.
- **cross-start — cross-end** - As linhas flexíveis são preenchidas com itens e colocadas no contêiner começando pelo lado de início cruzado do contêiner e indo em direção ao lado de fim cruzado.
- **cross size** - A largura ou altura de um item flexível, o que estiver na dimensão cruzada, é o tamanho cruzado do item. A propriedade do tamanho cruzado é a 'width' ou 'height', o que estiver na dimensão cruzada.

Propriedades do Flexbox

O Flexbox oferece várias propriedades que podem ser aplicadas tanto ao contêiner flexível (elemento pai) quanto aos itens flexíveis. Alguns exemplos de propriedades importantes incluem:

- **display: flex;** - Define um contêiner como um contêiner flexível.
- **flex-direction** - Especifica a direção principal do layout flexível (row, row-reverse, column, column-reverse).
- **flex-wrap** - Controla se os itens flexíveis podem quebrar para a próxima linha ou coluna.
- **justify-content** - Alinha os itens flexíveis ao longo do eixo principal do contêiner.
- **align-items** - Alinha os itens flexíveis ao longo do eixo cruzado do contêiner.
- **flex** - Define a capacidade de expansão e encolhimento dos itens flexíveis.

Propriedades para o Contêiner Pai (Flex Container)

As propriedades aplicadas ao contêiner pai, também conhecido como "flex container,"desempenham um papel fundamental na definição do layout flexível:

- ‘display: flex;‘ - Define o elemento como um contêiner flexível, iniciando um contexto de layout flexível.
- ‘flex-direction‘ - Determina a direção principal do layout flexível (row, row-reverse, column, column-reverse).
- ‘flex-wrap‘ - Controla como os itens flexíveis se comportam quando não há espaço suficiente no eixo principal (nowrap, wrap, wrap-reverse).

Propriedades para o Contêiner Pai (Flex Container)

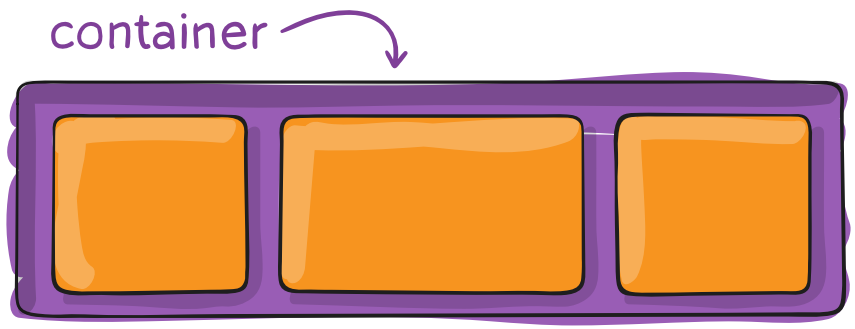
As propriedades aplicadas ao contêiner pai, também conhecido como "flex container," desempenham um papel fundamental na definição do layout flexível:

- ‘justify-content‘ - Define como os itens flexíveis são alinhados ao longo do eixo principal (flex-start, flex-end, center, space-between, space-around).
- ‘align-items‘ - Alinhamento dos itens flexíveis ao longo do eixo cruzado (flex-start, flex-end, center, baseline, stretch).
- ‘align-content‘ - Apenas relevante quando há várias linhas de itens flexíveis; define como o espaço é distribuído entre essas linhas (flex-start, flex-end, center, space-between, space-around, stretch).

Propriedade `display`

A propriedade `display` é fundamental no Flexbox, pois ela define se um elemento será um flex container e se comportará como um bloco ou em linha, dependendo do valor fornecido. Ela habilita um contexto de flex para todos os seus filhos diretos.
Por exemplo:

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



Propriedade ‘flex-direction‘

A propriedade ‘flex-direction‘ é usada para definir a direção principal do layout flexível em um flex container. Ela determina como os itens flexíveis são dispostos ao longo desse eixo principal. Aqui estão os valores possíveis para ‘flex-direction‘:

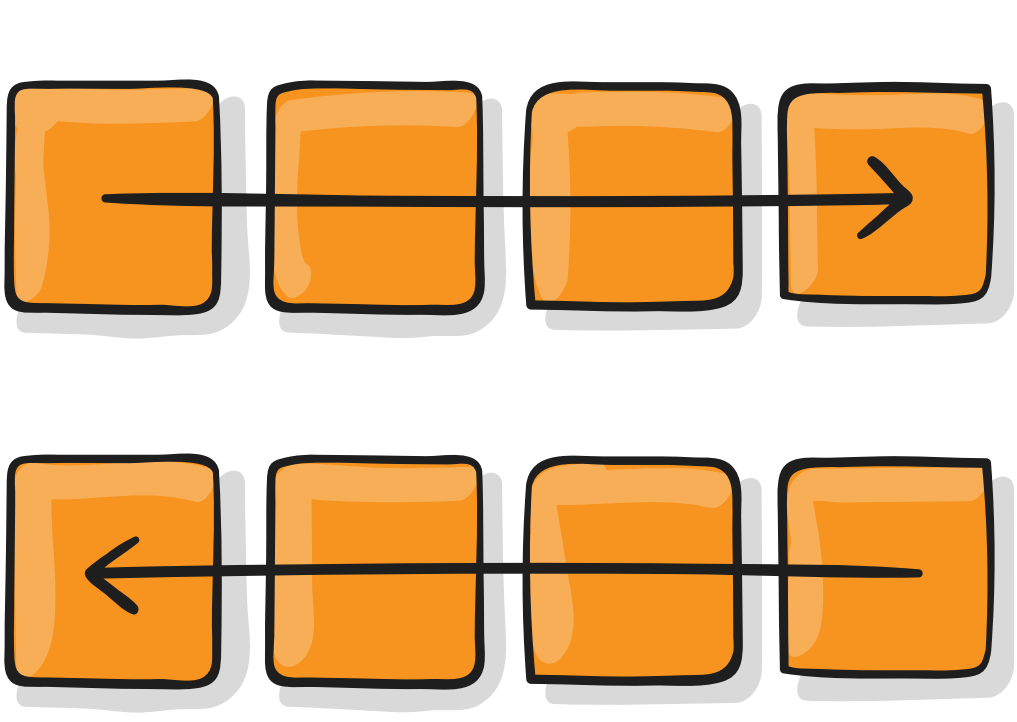
- **‘row‘:** Os itens flexíveis são dispostos na direção padrão do texto (horizontalmente da esquerda para a direita).
- **‘row-reverse‘:** Os itens flexíveis são dispostos na direção oposta à direção padrão do texto (horizontalmente da direita para a esquerda).
- **‘column‘:** Os itens flexíveis são dispostos verticalmente, de cima para baixo.
- **‘column-reverse‘:** Os itens flexíveis são dispostos verticalmente, de baixo para cima.

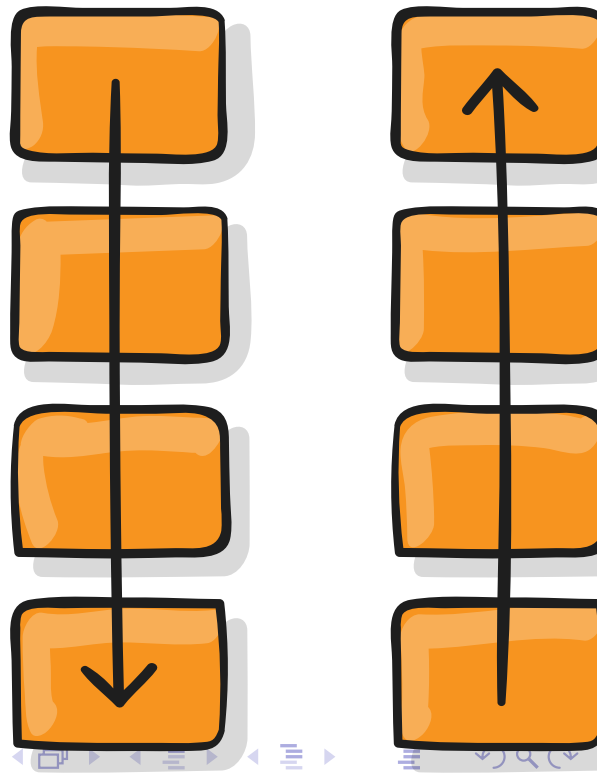
Propriedade flex-direction

Exemplo de uso:

```
.container {
  flex-direction: row | row-reverse | column | column-reve
}
```

A escolha do valor de ‘flex-direction‘ afeta a orientação dos itens flexíveis em relação ao contêiner flexível.





Exemplo de Uso da Propriedade ‘flex-direction‘

Considere o seguinte HTML e CSS:

```
<div class="container">
  <div class="item">Item 1</div>
  <div class="item">Item 2</div>
  <div class="item">Item 3</div>
  <div class="item">Item 4</div>
</div>
```

Exemplo de Uso da Propriedade ‘flex-direction‘

```
<style>
.container {
  display: flex;
  flex-direction: row; /* Itens são dispostos da esquerda p
}

.item {
  border: 1px solid #333;
  padding: 10px;
  margin: 5px;
}
</style>
```

Neste exemplo, definimos um contêiner com a classe ‘.container‘ e quatro itens com a classe ‘.item‘. Usamos ‘flex-direction: row;‘ para dispor os itens da esquerda para a direita ao longo do eixo principal.

Guia Completo de Flexbox

Se você deseja aprender mais sobre Flexbox e aprofundar seus conhecimentos, recomendamos o guia completo sobre Flexbox no site Origamid.
`https://origamid.com/projetos/flexbox-guia-completo/`

Guia Completo de Flexbox

Este guia oferece informações detalhadas e práticas sobre como usar o Flexbox para criar layouts flexíveis e responsivos em suas páginas da web. É uma ótima fonte de aprendizado.

Exemplo Flexbox

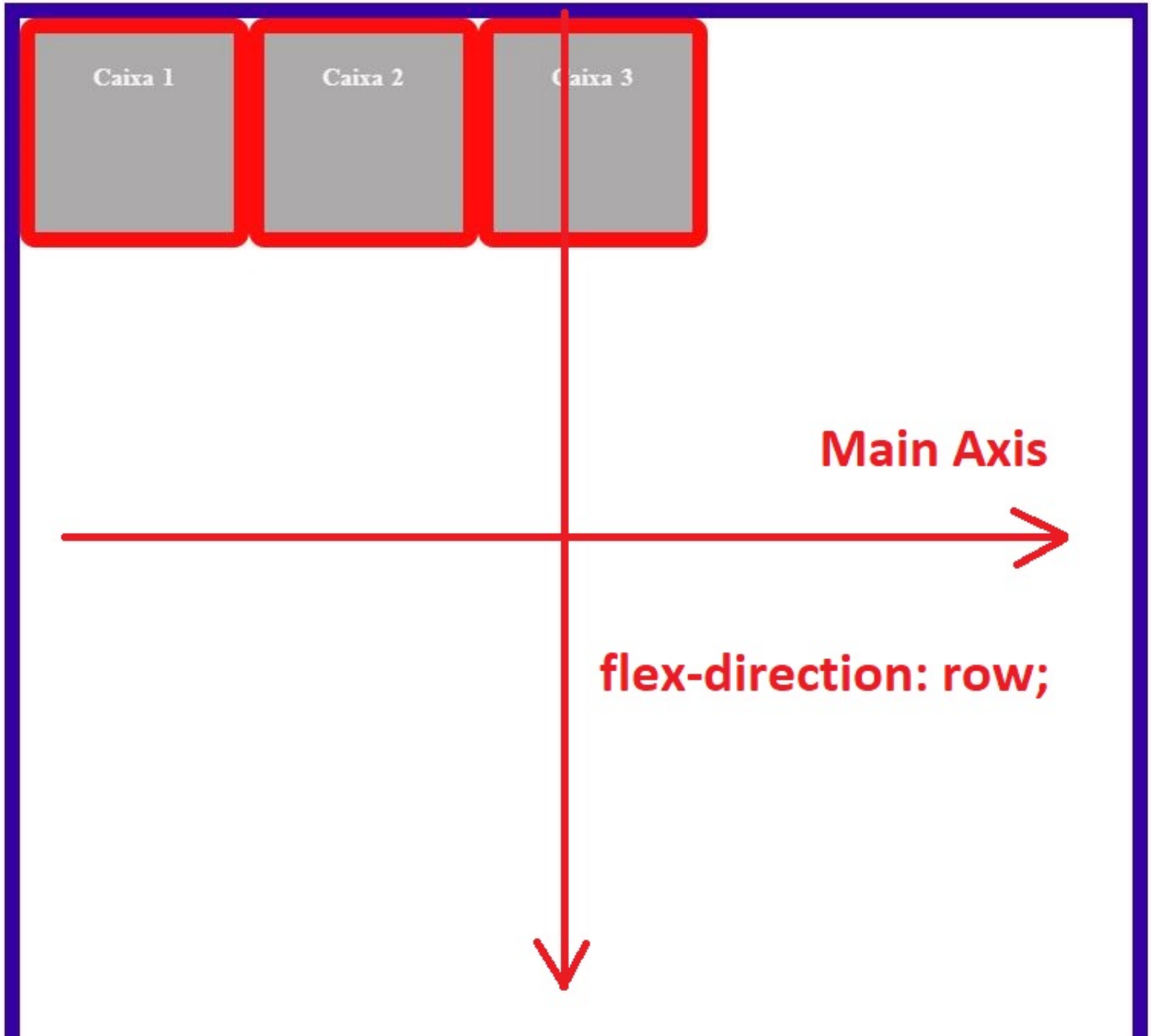
Download código base: `http://lamp/~heder/flex.zip`

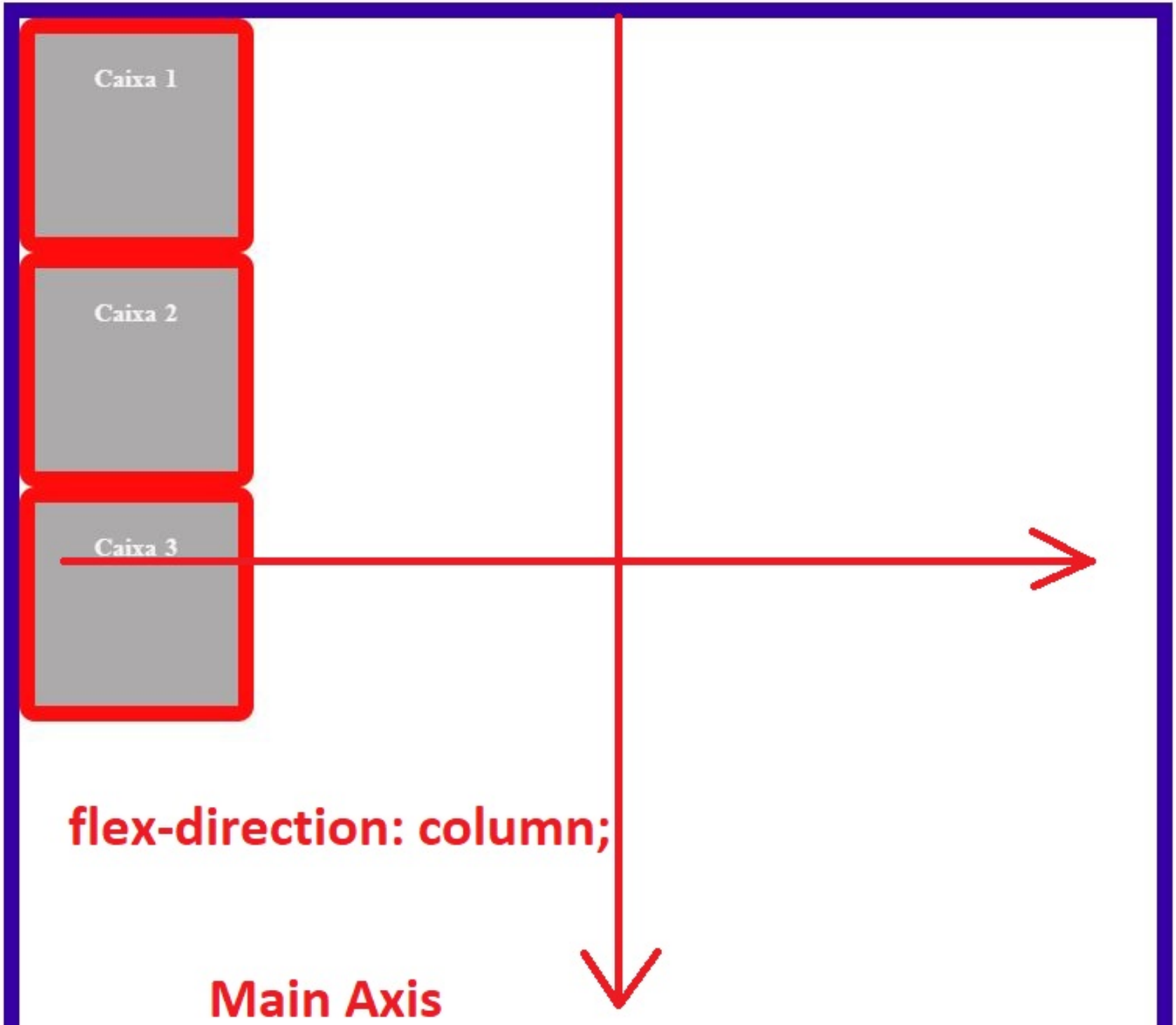
Flex Direction

O código CSS a seguir define um contêiner flexível com algumas propriedades específicas:

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

- **‘display: flex;‘:** Essa propriedade ‘display‘ é definida como ‘flex‘. Isso significa que o elemento com a classe ‘.container‘ será tratado como um contêiner flexível. Isso inicia um contexto de layout flexível para seus filhos diretos.
- **‘flex-direction: row;‘:** A propriedade ‘flex-direction‘ é definida como ‘row‘. Isso determina que os itens flexíveis dentro do contêiner serão dispostos na direção padrão do texto, que é da esquerda para a direita. Os itens serão alinhados horizontalmente.


Caixa 1
Caixa 2
Caixa 3
Main Axis
flex-direction: row;


Caixa 1
Caixa 2
Caixa 3
flex-direction: column;
Main Axis

Propriedade ‘justify-content‘

A propriedade ‘justify-content‘ é usada em um contêiner flexível para controlar como os itens flexíveis são alinhados ao longo do eixo principal (horizontalmente, se a direção principal for "row"e verticalmente, se for "column").

- **flex-start:** Alinhados no início do contêiner, deixando espaço em branco no final.
- **flex-end:** Alinhados no final do contêiner, deixando espaço em branco no início.
- **center:** Os itens são centralizados ao longo do eixo principal.
- **space-between:** Distribuídos uniformemente ao longo do eixo principal, com espaço igual entre eles, mas não no início e no final.
- **space-around:** Distribuídos uniformemente ao longo do eixo principal, com espaço igual em torno deles.
- **space-evenly:** Distribuídos uniformemente ao longo do eixo principal, com espaço igual entre eles, incluindo o início e o final.

Propriedade ‘justify-content‘

Exemplo de uso:

```
.container {
  display: flex;
  justify-content: center; /* ou outra opção */
}
```

A escolha da opção ‘justify-content‘ afeta a maneira como os itens flexíveis são alinhados horizontalmente ou verticalmente dentro do contêiner flexível.

Propriedade align-items

A propriedade ‘align-items‘ é usada para controlar como os itens flexíveis são alinhados ao longo do eixo cruzado (horizontalmente, se a direção principal for "column"e verticalmente, se for "row").

- **flex-start:** Alinhados no início do contêiner cruzado.
- **flex-end:** Alinhados no final do contêiner cruzado.
- **center:** Centralizados ao longo do eixo cruzado.
- **baseline:** Alinhados pela linha de base (ideal para alinhar o texto).
- **stretch:** Esticados para preencher o contêiner cruzado (padrão).

Propriedade align-items

Exemplo de uso:

```
.container {
  display: flex;
  justify-content: space-between;
  align-items: center; /* ou outra opção */
}
```

A escolha da opção ‘align-items‘ afeta a maneira como os itens flexíveis são alinhados verticalmente ou horizontalmente dentro do contêiner flexível.

Propriedade align-items

Faça o seguinte teste para verificar o uso de `baseline`:

```
.container {
    display: flex;
    justify-content: space-between;
    align-items: baseline;
}
.item-1{
    font-size: 24px;
}
```

Mude também o *flex-direction* para *column*, e verifique a mudança.

Flex Wrap

Replique as *divs* com mais seis caixas, para visualizar o efeito:

```
.container {
  display: flex;
  justify-content: space-between;
  align-items: baseline;
  flex-wrap: wrap;
}
```

Este código define um contêiner flexível com alinhamento entre os itens ao longo do eixo principal, alinhamento pela linha de base no eixo cruzado e permite que os itens quebrem para a próxima linha quando não couberem mais na largura disponível.

Propriedade align-content

A propriedade `align-content` é usada em um contêiner flexível para controlar como as linhas de itens flexíveis (quando ‘flex-wrap‘ está habilitado):

```
.container {
  display: flex;
  align-content: flex-start;
  flex-wrap: wrap;
}
```

Isso resulta em um layout onde as linhas de itens flexíveis são alinhadas no início do contêiner cruzado e os itens que não cabem na largura do contêiner são movidos para a próxima linha. Propriedade *gap* pode ser usada para espaço entre elementos.

Propriedade flex-grow

A propriedade flex-grow é usada para determinar como os itens flexíveis em um contêiner flexível devem se expandir em relação aos outros itens quando há espaço extra disponível ao longo do eixo principal.
- Um valor de ‘0‘ (padrão). - Valores maiores que ‘0‘ indicam a proporção de espaço extra que o item deve ocupar.

```
.item {
flex-grow: 1; /* Expandir igualmente com outros itens. */
}
.outro-item {
flex-grow: 2; /* Dobro da capacidade de expansão. */
}
```

A propriedade ‘flex-grow‘ é útil para criar layouts flexíveis onde alguns itens têm mais capacidade de expansão do que outros.

Propriedade ‘flex-shrink‘

A propriedade ‘flex-shrink‘ é usada para determinar como os itens flexíveis em um contêiner flexível devem encolher em relação aos outros itens quando há falta de espaço ao longo do eixo principal.
- Um valor de ‘1‘ (padrão).
- Valores maiores que ‘1‘ indicam a proporção em que o item deve encolher em relação aos outros, quando há falta de espaço.
- Um valor de ‘0‘ significa que o item não deve encolher, mesmo que haja falta de espaço.

Propriedade ‘flex-shrink‘

Exemplo de uso:

```
.container {
    display: flex;

}
.item-3 {
  flex-shrink: 1; /* O item pode encolher
                     igualmente com outros itens. */
}
```

A propriedade ‘flex-shrink‘ é útil para controlar como os itens flexíveis se comportam quando o espaço disponível é insuficiente.

Propriedade flex-basis

A propriedade flex-basis define o tamanho inicial de um item flexível antes de qualquer expansão ou encolhimento ocorrer.
- Pode ser definida com um valor fixo, porcentagem ou outras unidades de medida.
Exemplo de uso:

```
.item-3{
  flex-basis: 100px; /* Tamanho inicial com 100 px */
}
```

A propriedade ‘flex-basis‘ é útil para especificar o tamanho inicial de um item flexível em um contêiner flex.

Flex CSS

No CSS, podemos usar *shorthand* para definir várias propriedades Flexbox de uma só vez.

```
.item {
  flex: flex-grow flex-shrink flex-basis;
}
```

- flex-grow: Define o fator de crescimento. - flex-shrink: Define o fator de encolhimento. - flex-basis: Define o tamanho inicial.

```
.item-3 {
  flex: 1 0 100px; /* Cresce, não encolhe,
                      tamanho inicial de 100 pixels. */
}
```

Propriedade align-self

A propriedade align-self é usada para controlar o alinhamento vertical de um item flexível dentro de um contêiner flexível.
- Ela substitui o alinhamento definido pelo contêiner para um item específico.
Exemplo de uso:

```
.item-3 {
  align-self: flex-start; /* Alinha este item no
                              início do eixo cruzado. */
}
```

Propriedade ‘order‘

A propriedade ‘order‘ é usada para controlar a ordem de exibição dos itens flexíveis dentro de um contêiner flexível.
- Itens com valores menores de ‘order‘ aparecem antes dos itens com valores maiores (o valor padrão é 0).
Exemplo de uso:

```
.item-3 {
  order: -1; /* Este item aparecerá em primeiro. */
}
```

Flex Box adventure

https://codingfantasy.com/games/flexboxadventure/play