mdn web docs

This page was translated from English by the community. Learn more and join the MDN Web Docs community.

Conceitos básicos de flexbox

O *Flexible Box Module*, geralmente chamado de *flexbox*, foi projetado tanto como um modelo de *layout* unidimensional quanto como um método capaz de organizar espacialmente os elementos em uma interface, além de possuir capacidades avançadas de alinhamento. Este artigo fornece um resumo das principais funcionalidades do *flexbox*, as quais exploraremos com mais detalhes no restante deste guia.

Quando se descreve o flexbox como sendo unidimensional, enfatiza-se o fato de que ele lida com *o layout* em uma dimensão de cada vez - seja uma linha ou uma coluna. Isto pode ser comparado com o modelo bidimensional de CSS - Layout de Grade, que permite o controle simultâneo das colunas e linhas.

Os eixos do *flexbox*

Ao se utilizar o *flexbox*, é preciso ter em mente que todas as operações realizadas relacionam-se a dois eixos: o eixo principal e o eixo transversal. O eixo principal é definido através da propriedade `flex-direction` e o eixo transversal encontra-se na direção perpendicular a ele. Como esses eixos são as engrenagens fundamentais do flexbox é necessário compreender minuciosamente o seu funcionamento.

Eixo principal

Conforme descrito acima, a propriedade `flex-direction` define a direção do eixo principal e pode tem quatro valores possíveis:

- `row`

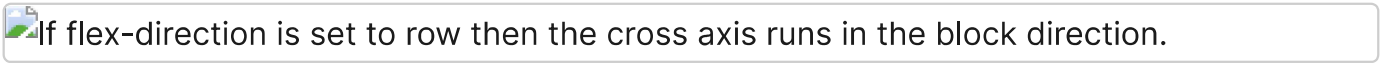
- `row-reverse`
- `column`
- `column-reverse`

Se o valor escolhido for `row` (linha) ou `row-reverse` (linha reversa), seu eixo principal se moverá ao longo da linha — na **direção inline**.

Se o valor escolhido for `column` (coluna) ou `column-reverse` (coluna reversa) e o eixo principal se moverá do topo até o fim da página — na **direção block**.

Eixo transversal

O eixo transversal é perpendicular ao eixo principal, logo, se a propriedade `flex-direction` estiver definida nas linhas, como `row` ou `row-reverse`, o eixo transversal estará na direção das colunas, como `column` ou `column-reverse`.



Se o eixo principal for definido nas colunas, como `column` ou `column-reverse`, então o eixo transversal estará na direção das linhas, como `row` ou `row-reverse`.

Compreender a diferença entre os eixos principal e perpendicular é o que importa quando começamos a observar o alinhamento ou justificação dos itens flexíveis (flex items); o *flexbox* possui propriedades que alinham e justificam o conteúdo ao longo de um eixo ou de outro.

Linhas de Início e Fim

Outra área crítica em termos de compreensão é como o Flexbox não faz nenhuma premissa sobre o modo de escrita do documento. No passado, o CSS utilizava fortemente os modos de escrita horizontal e da esquerda para a direita. Métodos modernos de layout abrangem uma variedade de modos de escrita e, portanto, não assumimos mais que uma linha de texto começará no canto superior esquerdo do documento e sigará para o lado direito, com novas linhas aparecendo uma após a as outra.

Essa discussão sobre a relação entre o *flexbox* e a especificação do modo de escrita será abordada um artigo posterior, contudo, a descrição a seguir explica brevemente porque não se fala sobre esquerda e direita/ acima e abaixo quando descreve-se a direção para a qual os elementos *flex* fluem.

Se o valor da propriedade `flex-direction` for `row (linha)`, considerando o estilo de escrita ocidental, a borda inicial do eixo principal estará localizada à esquerda e a borda final, à direita.

Considerando um idioma como o Árabe, que possui um estilo de escrita oriental, teremos o inverso: a borda inicia do eixo principal estará localizada à direita e a borda final, à esquerda.

Em ambos os casos, a borda inicial do eixo transversal está na parte superior do contêiner flex e a borda final, na parte inferior, visto que ambos os idiomas têm um estilo de de escrita horizontal.

Após um tempo de prática, pensar em termos de início e final ao invés de esquerda e direita se tornará natural e será útil ao lidar com outros métodos de layout, como CSS Grid, que seguem os mesmos padrões.

Contêiner *Flex*

A área de um documento que faz uso do *flexbox* é chamada de **contêiner *flex***. Para criar essa estrutura, define-se o valor da propriedade `display` do elemento representado pelo

contêiner como `flex` ou `inline-flex`. Desse modo, os elementos-filhos desse contêiner tornar-se-ão do tipo *flex*. Como todas as propriedades no CSS possuem valores padrão, ao criar um contêiner flex, os elementos nele contidos apresentarão o seguinte comportamento:

- Exibição em linha (o padrão do `flex-direction` é `row`).
- Alinhamento na borda inicial do eixo principal.
- Não há expansão no eixo principal, mas pode haver contração.
- Haverá expansão vertical para preencher a altura do eixo transversal.
- A propriedade `flex-basis` (en-US) estará definida como `auto`.
- A propriedade `flex-wrap` estará definida como `nowrap`.

O resultado final é que todos os elementos serão alinhados em uma linha, usando o tamanho do conteúdo como o tamanho no eixo principal. Se houver mais itens do que é possível caber no container, não haverá uma quebra de linha; ao invés disso, irão ultrapassar o limite horizontal da página. Se alguns elementos forem mais altos que outros, todos os itens se estenderão ao longo do eixo transversal para preencher seu tamanho total.

É possível conferir essas questões no exemplo prático abaixo. Tente editar ou adicionar mais itens para testar o comportamento inicial do Flexbox.


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

```
<div class="box">
  <div>One</div>
  <div>Two</div>
  <div>Three
    <br>has
    <br>extra
    <br>text
  </div>
</div>
```

Reset

Propriedade *`flex-direction`*

A propriedade `flex-direction` permite alterar a direção na qual os elementos flex serão exibidos ao longo do eixo principal. Definindo a propriedade `flex-direction` como `row-reverse` (linha reversa) ainda teremos os elementos dispostos em uma linha, entretanto, as linhas inicial e final serão trocadas.

Se mudarmos a `flex-direction` para a `column` (coluna), o eixo principal exibirá os elemento em uma coluna. Trocando o valor para `column-reverse` (coluna reversa) fará com que as linhas inicial e final sejam novamente trocadas.

No exemplo prático abaixo tem-se a propriedade `flex-direction` definida como `row-reverse`. Experimente os outros valores - `row`, `column` e `column-reverse` - para ver o que acontece com o conteúdo.


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

```
<div class="box">
  <div>One</div>
  <div>Two</div>
  <div>Three</div>
</div>
```

Reset

Quebra de linha com *"flex-wrap"*

Embora o *flexbox* seja um modelo unidimensional, é possível fazer com que elementos *flex* sejam agrupados em múltiplas linhas. Ao fazer isso, considera-se cada linha como um novo contêiner *flex*. Qualquer distribuição espacial ocorrerá ao longo essa linha, sem referência às linhas de ambos os lados. Para gerar a quebra automática das linhas adicione a propriedade `flex-wrap` com o valor `wrap`. Assim, se elementos forem muito grandes para serem exibidos em uma única linha, eles serão agrupados em outras linhas.

O exemplo prático abaixo contém elementos *flex* aos quais foi dada uma determinada largura, cuja soma totaliza um valor grande demais para caber no container. Visto que a propriedade `flex-wrap` está definida como `wrap`, os itens serão reorganizados em mais de uma linha. Trocando-se para `nowrap`, que também é o valor inicial, eles encolherão para caber no contêiner, porque estão usando valores iniciais de flexbox que permitem que os itens encolham. O uso do `nowrap` causaria um vazamento se os itens não encolhessem ou não diminuíssem o suficiente para caber.

Saiba mais sobre o empacotamento de itens flexíveis no guia Masterização de Empacotamento de Itens Flexíveis (en-US) (em Inglês).

Propriedade abreviada *flex-flow*

Pode-se combinar as propriedades `flex-direction` e `flex-wrap` de forma abreviada através da propriedade `flex-flow`. O primeiro valor especificado é o `flex-direction` e o segundo valor é o `flex-wrap`.

No exemplo prático abaixo, tente alterar o primeiro valor para um dos valores permitidos para a propriedade `flex-direction` - `row`, `row-reverse`, `column` ou `column-reverse`, e também altere o segundo para `wrap` e `nowrap`.

Expansão, encolhimento e tamanho dos elementos *flex*

Para ter mais controle sobre os elementos *flex* é possível alterá-los diretamente utilizando as três propriedades abaixo:

- `flex-grow` (en-US)
- `flex-shrink` (en-US)
- `flex-basis` (en-US)

Na presente seção, examinar-se-á brevemente tais propriedades. Para se aprofundar neste conteúdo sugere-se o tutorial Taxas de Controle de Elementos Flex ao Longo do Eixo Principal (en-US) (em inglês).

Antes que essas propriedades possam fazer sentido, é preciso compreender o conceito de **espaço disponível**. Quando se modifica o valor das propriedades acima, altera-se a forma que o espaço disponível é distribuído entre os elementos. Tal conceito de espaço disponível também é relevante quando se trata do alinhamento.

Conforme o exemplo abaixo, se houver três elementos com 100 pixels de comprimento em um contêiner de 500 pixels, então o espaço total necessário para acomodá-los será de 300 pixels. Desse modo, sobrarão 200 pixels de espaço útil. Se os valores iniciais não forem modificados, então o *flexbox* posicionará esse espaço após o último item.

Se for necessário que os elementos cresçam proprocionamente ou não e preencham o espaço disponível, deverá haver método que faça essa distribuição espacial entre eles, conforme será visto nas seções subsequentes.

Propriedade `flex-basis`

A propriedade `flex-basis` define o tamanho inicial dos elementos, em unidades de *pixel*, antes que o espaço remanescente seja redistribuído. O valor inicial desta propriedade é `auto` — neste caso o navegador observa se os itens possuem o mesmo tamanho. No exemplo acima, todos os itens têm uma largura de 100 pixels, que é utilizada como padrão na propriedade `flex-basis`.

Se os elementos não possuírem um tamanho padrão, então as dimensões dos seus conteúdos (imagem, texto, etc) serão passadas como parâmetro para propriedade flex-basis. É por isso que quando escreve-se `display: flex` no elemento-pai para criar o contêiner, todos os elementos-filhos se organizam em linha e ocupam apenas o espaço necessário para exibir seu conteúdo.

Propriedade `flex-grow`

Com a propriedade `flex-grow` definida como um inteiro positivo, os elementos *flex* podem crescer ao longo do eixo principal, a partir do valor mínimo estabelecido no `flex-basis`. Isto fará com que o elemento se estique e ocupe qualquer espaço disponível nesse eixo ou uma proporção dele, caso outros elementos-irmãos também possam crescer.

Atribuir o valor 1 à propriedade `flex-grow` fará com que o espaço disponível no contêiner *flex* seja igualmente distribuído entre todos os elementos do exemplo acima. Logo, os elementos-filhos irão se expandir para preencher o contêiner no sentido do eixo principal.

Como visto no parágrafo anterior, a propriedade `flex-grow` pode ser empregada para distribuir o espaço proporcionalmente entre os elementos de um contêiner, contudo, se atribuirmos ao primeiro elemento o valor 2 e 1 aos elementos restantes, duas partes serão dadas ao primeiro elemento (100px de 200px totais) e uma parte para cada um dos outros dois elementos (50px de 200px totais).

Propriedade `flex-shrink`

Enquanto a propriedade `flex-grow` permite aumentar a largura dos elementos dentro do contêiner para completar o espaço disponível no eixo principal, a propriedade `flex-shrink` faz o oposto, controlando a redução dos mesmos. Caso não haja espaço suficiente para acomodar todos os elementos e o valor da propriedade `flex-shrink` seja um inteiro positivo, a largura pode ser reduzida a um valor menor do que a definida na propriedade `flex-basis`. Assim como na propriedade flex-grow, diferentes valores podem ser atribuídos a um elemento de modo que ele encolha mais do que os outros - um elemento cuja propriedade `flex-shrink` receba um valor inteiro maior irá diminuir mais do que os seus irmão que tenham valores menores.

O tamanho mínimo do elemento será levado em consideração ao se calcular a quantidade real de encolhimento que ocorrerá, o que significa que a propriedade flex-shrink se comporta de modo potencialmente menos consistente do que a propriedade flex-grow. Examinar-se-á mais detalhadamente o funcionamento desse algoritmo no artigo Taxas de Controle de Elementos Flex ao Longo do Eixo Principal.

> **Nota:** Note que os valores para as propriedades `flex-grow` e `flex-shrink` são proporcionais. Particularmente, se tivermos todos os nossos elementos definidos como flex: `1 1 200px` e então quisermos que um deles cresça o dobro, temos de definir o elemento como flex: `2 1 200px`. Entretanto, podemos escrever flex: `10 1 200px` e flex: `20 1 200px` se quisermos.

Abreviatura para os valores das propriedades flex

As propriedades `flex-grow`, `flex-shrink`, and `flex-basis` raramente são empregas de forma individual. Usualmente, elas são combinadas através da propriedade de abreviação `flex`. A abreviatura `flex` permite definir os três valores na seguinte ordem: `flex-grow`, `flex-shrink`, `flex-basis`.

O exemplo prático abaixo permite que sejam testados diferentes valores com a propriedade de abreviação `flex`; lembre-se que o primeiro campo corresponde à propriedade `flex-grow`, onde um valor inteiro e positivo faz-se o elemento crescer. O segundo campo é a propriedade `flex-shrink` e, ao contrário do anterior, um valor inteiro e positivo pode fazer os elementos encolherem, mas somente se o seu comprimento total ultrapassar o limite horizontal do contêiner, no sentido do eixo principal. O último campo contém a propriedade `flex-basis`, que define o valor base, em unidades de *pixel*, para aumentar e diminuir o elemento quando da aplicação das propriedades anteriores.

Há ainda alguns valores de abreviação predefinidos, que cobrem a maioria dos casos de uso. São aplicados com frequência em turoriais e, normalmente, suprem todas as necessidades práticas. Os valores predefinidos podem ser vistos abaixo:

- `flex: initial`
- `flex: auto`
- `flex: none`

- `flex: <positive-number>`

Definir `flex: initial` reseta os elementos para valores-padrão do Flexbox, sendo equivale a `flex: 0 1 auto`. Neste ultimo caso, o valor da propriedade `flex-grow` é 0, então os elementos não irão crescer mais do que o tamanho-base definido na propriedade `flex-basis`. o valor da propriedade `flex-shrink` é 1, indicando que o elemento pode ser reduzido caso seja necessário, para evitar que o limite do contêiner seja ultrapassado. Por fim, o valor da propriedade `flex-basis` é `auto` e assim será usad o tamanho mínimo necessário para preencher a dimensão do eixo principal.

Definir `flex: auto` é equivalente a `flex: 1 1 auto`. Essa configuração é semelhante a `flex:initial`, mas nesse caso os elementos podem aumentar para preencher o contêiner ou diminuir se necessário, para evitar o transbordamento lateral da tela.

Definir `flex: none` irá criar um elemento flex totalmente inflexível, sendo o equivalente a escrever `flex: 0 0 auto`. O elementos não poderão crescer ou diminuir, mas serão criados usando o *flexbox* com a propriedade `flex-basis` com o valor `auto`.

Outra abreviação normalmente vista em tutoriais é `flex: 1` ou `flex: 2` e assim por diante, o que equipara-se a `flex: 1 1 0`. Os elementos podem crescer ou diminuir a partir da propriedade `flex-basis` com valor nulo.

Teste essas formas abreviadas no exemplo prático abaixo:

Alinhamento, justificação e distribuição de espaço livre entre os elementos

Um atributo chave do *flexbox* é a capacidade de alinhar e justificar os elementos *flex* nos eixos principal e transversal e distribuir o espaço entre eles.

Propriedade `align-items`

A propriedade `align-items` irá alinhar os elementos no eixo transversal.

O valor inicial desta propriedade é `stretch` e é por essa razão que, por padrão, os elementos *flex* se estendem até a maior altura. De fato, eles se esticam para preencher o contêiner flex - o item mais alto define a altura deste.

Pode-se definir a propriedade `align-items` como `flex-start`, de modo que os elementos fiquem alinhados com topo do contêiner, `flex-end` para alinhá-los a partir da base ou

`center`, para que o alinhamento seja centralizado.

Teste essa propriedade e seus possíveis valores no exemplo prático abaixo — colocou-se uma determinada altura no contêiner flex, de modo que se perceba como os elementos podem ser movidos no interior do mesmo. Veja o que acontece ao definir cada um dos possíveis valores da propriedade **align-items**:

- `stretch`
- `flex-start`
- `flex-end`
- `center`

Propriedade `justify-content`

A propriedade `justify-content` (en-US) é empregada para alinhar os elementos ao longo do eixo principal, cuja direção (`row` ou `column`) é definida a partir da propriedade `flex-direction`. O valor inicial é `flex-start`, que alinha os elementos rente à borda esquerda do contêiner, mas também pode ser definido como `flex-end`, que resulta em um alinhamento oposto, rente à borda direita do contêiner, ou `center`, para alinhá-los ao centro.

O valor `space-between` pode ser usado pode ser usado para ocupar todo o espaço livre após a disposição dos itens e dividí-lo igualmente entre os itens, para que haja a mesma quantidade de espaço entre cada elemento. Para gerar uma quantidade igual de espaço à direita e à esquerda, usa-se o valor `space-around`.

Tente os seguintes valores da propriedada `justify-content` no exemplo prático abaixo:

- `stretch`
- `flex-start`
- `flex-end`
- `center`
- `space-around`
- `space-between`

No artigo Alinhando Elementos em um Contêiner Flex (em inglês) tais propriedades serão abordadas mais detalhadamente, de modo a compreender melhor o seu funcionamento. Contudo, os exemplos simples abordados aqui serão úteis na maioria dos casos.

Próximos passos

Após ler este artigo, você deve ser capaz de compreender as características básicas do Flexbox. No próximo artigo, iremos examinar como essa especificação se relaciona com outras partes do CSS (en-US) (em inglês).