

# CSS Flexbox: Layout Flexível e Responsivo

O Flexbox é um módulo de layout unidirecional no CSS3, projetado para ajudar a alinhar, justificar e distribuir itens em um contêiner.

## 1. Ativando o Flexbox

A primeira etapa é definir o elemento pai como um **Contêiner Flex (Flex Container)**.

Propriedade	Exemplo de Declaração	Descrição
display	<code>.container { display: flex; }</code>	Define o contêiner como flexível.
display	<code>.container { display: inline-flex; }</code>	Define um contêiner flexível que se comporta como um elemento inline.

## 2. Propriedades do Contêiner Flex

Estas propriedades controlam a direção principal, o envolvimento das linhas e o alinhamento geral dos itens.

### 2.1. Direção e Fluxo (Main Axis)

Controla a direção do **eixo principal** (`main axis`).

Propriedade	Exemplo de Declaração	Descrição dos Valores
flex-direction	<code>.c { flex-direction: row; }</code>	(Padrão) Itens dispostos horizontalmente, da esquerda para a direita.
flex-direction	<code>.c { flex-direction: column; }</code>	Itens dispostos verticalmente, de cima para baixo.
flex-direction	<code>.c { flex-direction: row-reverse; }</code>	Itens horizontalmente, da direita para a esquerda.
flex-direction	<code>.c { flex-direction: column-reverse; }</code>	Itens verticalmente, de baixo para cima.
flex-wrap	<code>.c { flex-wrap: wrap; }</code>	Permite que os itens quebrem para a próxima linha se não houver espaço.
flex-wrap	<code>.c { flex-wrap: nowrap; }</code>	(Padrão) Todos os itens na mesma linha.
flex-flow	<code>.c { flex-flow: row wrap; }</code>	<i>Shorthand</i> para <code>flex-direction</code> e <code>flex-wrap</code> .

## 2.2. Alinhamento e Distribuição

Controla como o espaço livre é distribuído ao longo do **eixo principal** (`main axis`) e do **eixo transversal** (`cross axis`).

Propriedade	Exemplo de Declaração	Descrição dos Valores
<code>justify-content</code>	<code>.c { justify-content: center; }</code>	Alinha os itens ao centro do eixo principal.
<code>justify-content</code>	<code>.c { justify-content: space-between; }</code>	Distribui o espaço uniformemente <i>entre</i> os itens (sem margem nas bordas).
<code>justify-content</code>	<code>.c { justify-content: space-around; }</code>	Distribui o espaço uniformemente <i>ao redor</i> de cada item.
<code>justify-content</code>	<code>.c { justify-content: flex-start; }</code>	Alinha os itens ao início do eixo principal.
<code>align-items</code>	<code>.c { align-items: center; }</code>	Alinha os itens ao centro do eixo transversal (verticalmente em <code>row</code> ).
<code>align-items</code>	<code>.c { align-items: stretch; }</code>	(Padrão) Estica os itens para preencher o contêiner ao longo do eixo transversal.
<code>align-content</code>	<code>.c { align-content: space-between; }</code>	Alinha as <i>linhas</i> do contêiner (quando <code>flex-wrap: wrap</code> está ativo).
<b>gap</b>	<code>.c { gap: 10px; }</code>	Define o espaçamento entre itens (linhas e colunas).

## 3. Propriedades dos Itens Flex

Estas propriedades controlam o comportamento individual dos elementos filhos dentro do contêiner flex.

### 3.1. Ordenação e Alinhamento Individual

Propriedade	Exemplo de Declaração	Descrição
<code>order</code>	<code>.item-3 { order: 1; }</code>	Define a ordem visual do item (padrão é 0). Itens com <code>order</code> menor aparecem primeiro.
<code>align-self</code>	<code>.item-1 { align-self: flex-end; }</code>	Sobrescreve <code>align-items</code> do contêiner para alinhar um item específico no eixo transversal.

### 3.2. Flexibilidade e Dimensionamento

Controla como os itens crescem, encolhem e qual seu tamanho base.

Propriedade	Exemplo de Declaração	Descrição
<b>flex-grow</b>	<code>.item { flex-grow: 1; }</code>	Um fator numérico que indica o quanto o item deve crescer para ocupar o espaço extra disponível.
<b>flex-shrink</b>	<code>.item { flex-shrink: 0; }</code>	Um fator numérico que indica o quanto o item pode encolher se não houver espaço (padrão é 1).
<b>flex-basis</b>	<code>.item { flex-basis: 150px; }</code>	Define o tamanho inicial do item <i>antes</i> que o <b>grow</b> ou <b>shrink</b> sejam aplicados.
<b>flex</b>	<code>.item { flex: 1 1 auto; }</code>	<i>Shorthand</i> para <b>flex-grow</b> , <b>flex-shrink</b> e <b>flex-basis</b> .
<b>flex</b>	<code>.item { flex: 1; }</code>	(Comum) Equivalente a <code>flex: 1 1 auto;</code> (cresce, encolhe, base automática).
<b>flex</b>	<code>.item { flex: none; }</code>	Equivalente a <code>flex: 0 0 auto;</code> (não cresce nem encolhe).

## Exemplos de Código Completos

### Exemplo 1: Centrando Conteúdo

Um padrão comum é centralizar um elemento perfeitamente no meio da tela.

html

```
<div class="container-centro">
  <div class="item-centro">Conteúdo Centralizado</div>
</div>
```

css

```
.container-centro {
  display: flex;
  justify-content: center; /* Alinha horizontalmente */
  align-items: center;     /* Alinha verticalmente */
  height: 100vh;          /* Define a altura total da tela */
}

.item-centro {
  padding: 20px;
  background-color: lightblue;
}
```

### Exemplo 2: Layout de Colunas Responsivas com `flex-wrap`

Criando um layout de galeria que se ajusta automaticamente.

html

```
<div class="galeria">
  <div class="foto">1</div>
  <div class="foto">2</div>
  <div class="foto">3</div>
  <div class="foto">4</div>
  <div class="foto">5</div>
</div>
```

CSS

```
.galeria {
  display: flex;
  flex-wrap: wrap; /* Permite que os itens quebrem para a próxima linha */
  gap: 10px;       /* Espaçamento entre as fotos */
}

.foto {
  /* flex-basis: 150px define um tamanho mínimo antes de distribuir espaço */
  /* flex-grow: 1 permite que eles cresçam para preencher a linha */
  flex: 1 1 150px;
  background-color: #f0f0f0;
  padding: 20px;
  text-align: center;
  min-width: 100px; /* Garante que não encolham muito */
}
```