Flexbox in CSS

Flexbox (Flexible Box Layout) is a powerful layout system in CSS that allows you to **align, space, and distribute elements** easily — especially when building responsive layouts.

It's ideal for **one-dimensional layouts** (either a row or a column).

Getting Started

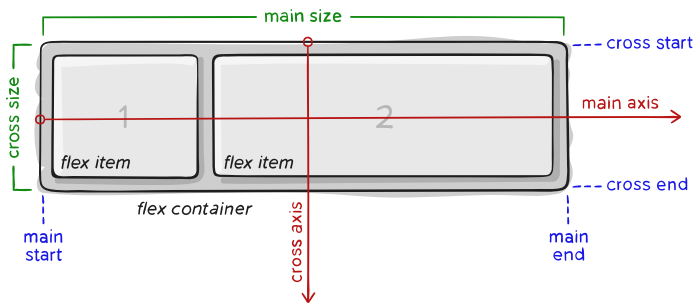
To use Flexbox, set the parent container's `display` to `flex`:

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

Now, all **direct children** of `.container` become **flex items**.

Main Concepts

Term	Description
Main Axis	The primary direction (`row` by default)
Cross Axis	Perpendicular to main axis
Flex Container	The parent element with `display: flex`
Flex Items	The children inside the container



Flex Direction

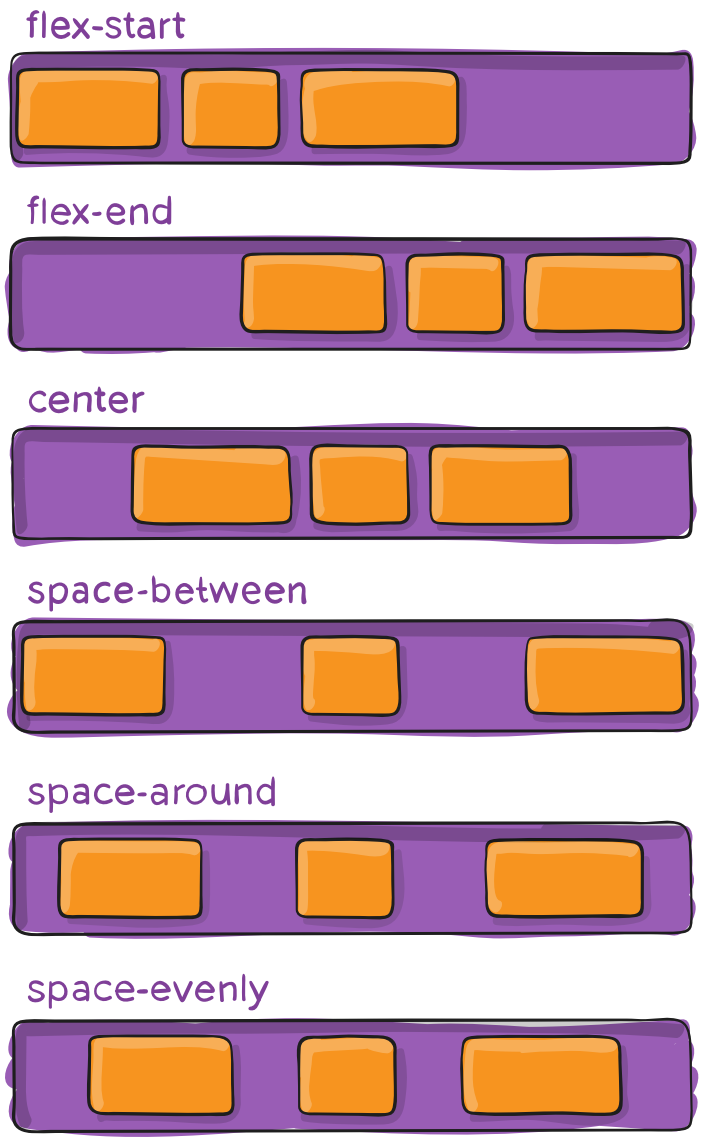
Controls the direction of flex items.

```
.container {
  display: flex;
  flex-direction: row;      /* default */
  flex-direction: row-reverse;
  flex-direction: column;
  flex-direction: column-reverse;
}
```

Justify Content (Main Axis Alignment)

Controls how items are **aligned along the main axis** (horizontal by default).

```
.container {
  justify-content: flex-start;   /* default */
  justify-content: flex-end;
  justify-content: center;
  justify-content: space-between;
  justify-content: space-around;
  justify-content: space-evenly;
}
```

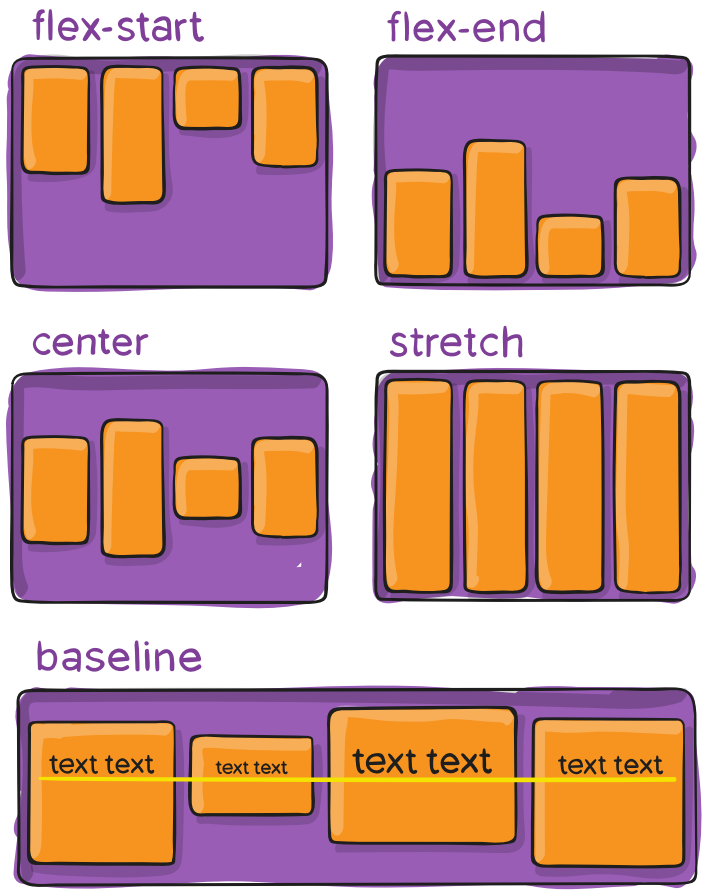


Align Items (Cross Axis Alignment)

Controls how items are **aligned on the cross axis** (vertical by default).

```
.container {
  align-items: stretch;     /* default */
  align-items: flex-start;
  align-items: flex-end;
  align-items: center;
```

```
  align-items: baseline;
}
```



Align Self

Allows individual items to override `align-items`.

```
.item {
  align-self: flex-end;
}
```

Flex Wrap

By default, items try to fit into a single line. Use `flex-wrap` to wrap them:

```
.container {
  flex-wrap: wrap;
  flex-wrap: nowrap;       /* default */
  flex-wrap: wrap-reverse;
}
```

Gap (Spacing Between Items)

```
.container {
  gap: 20px;
}
```

This replaces the need for margins between flex items.

Flex Grow, Shrink, Basis

Control how items grow, shrink, or have an initial size:

```
.item {
  flex-grow: 1;    /* takes remaining space */
  flex-shrink: 1;  /* shrink if needed */
  flex-basis: 200px; /* default size */
}
```

Shorthand:

```css
.item {
  flex: 1 1 200px;
}
```

Example Layout

```html
<div class="container">
  <div class="item">One</div>
  <div class="item">Two</div>
  <div class="item">Three</div>
</div>
```

```css
.container {
  display: flex;
  justify-content: space-between;
  align-items: center;
  gap: 10px;
}
.item {
  background: lightgray;
  padding: 20px;
  flex: 1;
}
```

Summary

- `display: flex` turns a container into a Flexbox layout.
- Use `justify-content`, `align-items`, and `flex-direction` to control layout flow.
- `flex` shorthand (`grow shrink basis`) gives you fine-grained sizing control.
- Use `gap` instead of margins for consistent spacing.