**CSS Flexbox — Complete Guide**

Flexbox (**Flexible Box Layout**) is a CSS layout model that makes it easier to arrange elements in rows or columns, and to align and distribute space between them.  
It is especially useful for **responsive designs** where elements need to adjust to different screen sizes.

**1. How to Enable Flexbox**

To use Flexbox, set the **container’s** display property to flex or inline-flex.

css

CopyEdit

.container {

display: flex; /\* block-level flex container \*/

/\* or \*/

display: inline-flex; /\* inline-level flex container \*/

}

**2. Understanding Axes in Flexbox**

* **Main Axis** — the primary direction items flow.
  + Row: main axis is **horizontal**.
  + Column: main axis is **vertical**.
* **Cross Axis** — perpendicular to the main axis.
  + Row: cross axis is **vertical**.
  + Column: cross axis is **horizontal**.

**3. Flex Container Properties**

**a) flex-direction**

Defines the direction of the main axis.

| **Value** | **Items flow…** |
| --- | --- |
| row | left → right |
| row-reverse | right → left |
| column | top → bottom |
| column-reverse | bottom → top |

css

CopyEdit

.container {

flex-direction: row;

}

**b) flex-wrap**

Controls whether flex items wrap onto multiple lines.

| **Value** | **Meaning** |
| --- | --- |
| nowrap | All items in one line (default). |
| wrap | Items wrap to new lines. |
| wrap-reverse | Items wrap in reverse order. |

css

CopyEdit

.container {

flex-wrap: wrap;

}

**c) flex-flow**

Shorthand for flex-direction + flex-wrap.

css

CopyEdit

.container {

flex-flow: row wrap;

}

**d) justify-content**

Aligns items along the **main axis**.

| **Value** | **Effect** |
| --- | --- |
| flex-start | Items at the start. |
| flex-end | Items at the end. |
| center | Items centered. |
| space-between | Equal space *between* items. |
| space-around | Equal space *around* items. |
| space-evenly | Equal space *between and around* items. |

css

CopyEdit

.container {

justify-content: center;

}

**e) align-items**

Aligns items along the **cross axis**.

| **Value** | **Row effect** | **Column effect** |
| --- | --- | --- |
| flex-start | Top | Left |
| flex-end | Bottom | Right |
| center | Middle | Center |
| stretch | Fill height | Fill width |
| baseline | Align text lines | Align text lines |

css

CopyEdit

.container {

align-items: center;

}

**f) align-content**

Aligns multiple lines of items (if wrapping) along the **cross axis**.  
*Only applies when there are multiple rows or columns of flex items.*

css

CopyEdit

.container {

align-content: space-between;

}

**4. Flex Item Properties**

These apply **to the items inside the flex container**, not the container itself.

**a) order**

Changes the order in which items appear.

css

CopyEdit

.item {

order: 2; /\* Higher numbers appear later \*/

}

**b) flex-grow**

Defines how much space an item takes compared to others.

css

CopyEdit

.item {

flex-grow: 1; /\* Shares extra space equally \*/

}

**c) flex-shrink**

Defines how much an item shrinks when space is tight.

css

CopyEdit

.item {

flex-shrink: 0; /\* Item won’t shrink \*/

}

**d) flex-basis**

Sets the starting size of the item before growing/shrinking.

css

CopyEdit

.item {

flex-basis: 200px;

}

**e) flex**

Shorthand for flex-grow, flex-shrink, and flex-basis.

css

CopyEdit

.item {

flex: 1 1 200px;

}

**f) align-self**

Overrides align-items for a single item.

css

CopyEdit

.item {

align-self: flex-end;

}

**5. Example — Center Everything**

css

CopyEdit

.container {

display: flex;

justify-content: center; /\* center horizontally \*/

align-items: center; /\* center vertically \*/

height: 100vh;

}

**Using the flex Shorthand Property in CSS**

The flex property is a shorthand that allows you to set three Flexbox item properties in one line:

* **flex-grow** — Defines how much the flex item should grow relative to others when extra space is available.
* **flex-shrink** — Defines how much the flex item should shrink relative to others when space is limited.
* **flex-basis** — Defines the initial main size of the flex item before any growing or shrinking occurs.

**Syntax:**

css

CopyEdit

flex: <flex-grow> <flex-shrink> <flex-basis>;

**Example**

In the example below, two flex items are assigned different growth and shrink rates while sharing the same starting size.

css

CopyEdit

#box-1 {

flex: 2 2 150px; /\* grow: 2, shrink: 2, basis: 150px \*/

}

#box-2 {

flex: 1 1 150px; /\* grow: 1, shrink: 1, basis: 150px \*/

}

**How It Works**

* **Initial Size:** Both boxes start with a width of 150px (flex-basis value).
* **Growing:** If the flex container has extra space, #box-1 grows twice as fast as #box-2 because its flex-grow value is 2 compared to 1.
* **Shrinking:** If the container is smaller than 300px (the combined basis sizes), #box-1 shrinks at twice the rate of #box-2 because its flex-shrink value is 2 compared to 1.

**Result:**  
When space changes, #box-1 adjusts more aggressively than #box-2, both when expanding and when contracting.