

## CSS Flexbox – Layout, Order & Alignment

### 1. Understanding Parent vs Child Properties

Flexbox has two kinds of properties:

- **Parent (Container)** – controls *how items behave together*.
- **Child (Item)** – controls *how an individual item behaves*.

Always check if the property applies to **container** or **item**, or it won't work.


---

### 2. The order Property (Child Property)

**Purpose:** Changes the *visual order* of flex items, without changing the HTML.

- Default: order: 0
- Items are arranged by their HTML order unless you manually change it.

```
.item {  
  order: 2; /* Moves this item after those with order 0 or 1 */  
}
```

 **Concept:** Higher order = further to the end of the main axis.  
E.g., if you set order: 10 for green, it'll appear last.


---

### 3. The flex-wrap Property (Parent Property)

**Purpose:** Controls whether flex items stay in one line or wrap onto multiple lines.

Value	Description
nowrap	Default. Items stay in one line, may overflow container.
wrap	Items move to the next line if no space left.
wrap-reverse	Same as wrap, but items wrap from bottom to top (reverse order).

```
.container {  
  flex-wrap: wrap;  
}
```

 **Tip:** Use wrap for responsive layouts — prevents items from disappearing off-screen.

---

### 4. The justify-content Property (Parent Property)

**Purpose:** Aligns and distributes items **along the main axis** (horizontal by default).

Value	What It Does
flex-start	Items start from the left.
flex-end	Items move to the right end.
center	Centers items along the main axis.
space-between	Equal gaps <i>between</i> items; first & last stick to edges.
space-around	Equal gaps around each item (outer gaps smaller).
space-evenly	Perfectly equal gaps on all sides.
<pre>.container {   justify-content: space-evenly; }</pre>	

💡 **Shortcut to center horizontally:**  
 justify-content: center;

## 5. The align-items Property (Parent Property)

**Purpose:** Aligns items **along the cross-axis** (vertical by default).

Value	What It Does
flex-start	Aligns items at the top.
flex-end	Aligns items at the bottom.
center	Vertically centers items.
baseline	Aligns items based on text baseline.
stretch	Stretches items to fill container height.
<pre>.container {   align-items: center; }</pre>	

⚠️ Works best when the container has a fixed height (e.g., height: 70vh).

## 6. The align-self Property (Child Property)

**Purpose:** Overrides align-items for a specific item.

```
.item.special {
```

```
align-self: flex-start;
}
```

➡ Use this when you want one item to break from the pack and align differently.

---

## 7. The align-content Property (Parent Property)

**Purpose:** Controls spacing between *rows* of wrapped content (only works when flex-wrap: wrap is enabled).

Value	What It Does
flex-start	All rows packed at the top.
flex-end	All rows packed at the bottom.
center	Rows grouped in the middle.
space-between	Rows evenly distributed, first & last touch edges.
space-around	Equal space around each row.
stretch	Rows stretch to fill container.

```
.container {
  flex-wrap: wrap;
  align-content: space-around;
}
```

### ✖ Difference from align-items:

- align-items = aligns *items* in one line.
  - align-content = aligns *multiple lines* when wrapping.
- 

## 8. Main Axis vs Cross Axis Recap

Flex Direction	Main Axis	Cross Axis
row	Left → Right	Top → Bottom
column	Top → Bottom	Left → Right

- justify-content works **along main axis**
  - align-items & align-content work **along cross axis**
- 

## 9. Resources & Practice

- 📄 **Cheat Sheet:** CSS Tricks Flexbox Guide
  - 🎮 **Practice Game:** [Flexbox Froggy](#)
    - Choose **Intermediate** level.
    - Use what you learned: justify-content, align-items, flex-wrap, etc.
    - Goal: Move frogs to their lily pads with correct CSS.
- 

## 10. Key Takeaways

- Flexbox gives **automatic alignment and spacing** — no more float madness.
- Parent controls *group behavior*, child controls *individual quirks*.
- Don't memorize; **understand what's possible** and look up syntax when needed.
- Practice layouts interactively — it'll stick better than reading specs.