Summary

* ChatGPT

**Key Properties:**

1. **flex-direction**: Defines the direction of items (e.g., row or column).
2. **justify-content**: Aligns items along the main axis (horizontal), controlling spacing.
3. **align-items**: Aligns items along the cross axis (vertical).
4. **flex-wrap**: Allows items to wrap onto new lines on smaller screens.

**Example Layout:**

* A container holds three boxes arranged in a row. On smaller screens, the layout switches to a vertical stack using media queries.

**How Flexbox Helps:**

* **Easier alignment** of content both horizontally and vertically.
* **Responsive design**: Items adjust automatically based on screen size without needing complex CSS.
* Gemini

**Key Concepts:**

* **Flex Container:** The parent element containing flex items.
* **Flex Items:** Child elements within the container.
* **Flex Properties:** Control the arrangement of flex items.

**Common Flexbox Properties:**

* display: flex; sets the container as a flex container.
* justify-content controls horizontal alignment.
* align-items controls vertical alignment.
* flex-direction sets the layout direction (row, column).

**Example:**

HTML

<div class="container">

<div class="item">Item 1</div>

<div class="item">Item 2</div>

<div class="item">Item 3</div>

</div>

Use code [with caution.](/faq#coding)

CSS

.container {

display: flex;

justify-content: space-between;

align-items: center;

}

Use code [with caution.](/faq#coding)

This creates a container with three horizontally spaced items, centered vertically.

**Key Points:**

* Use Flexbox for responsive layouts.
* Experiment with different properties to achieve desired arrangements.

Sources and related content