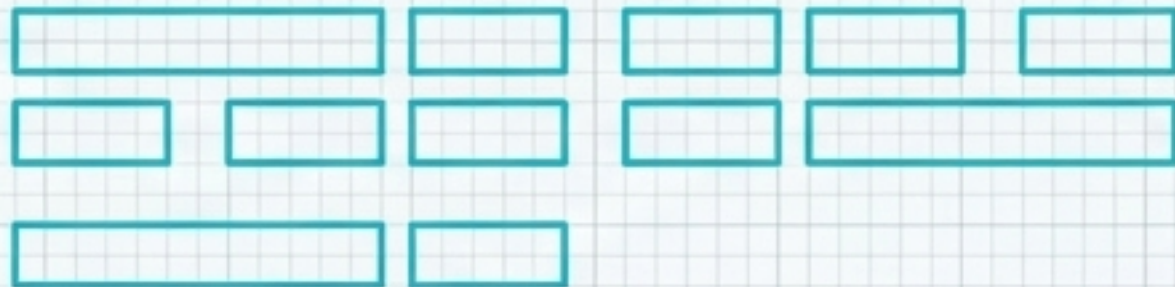
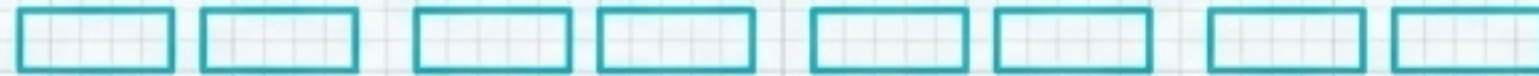


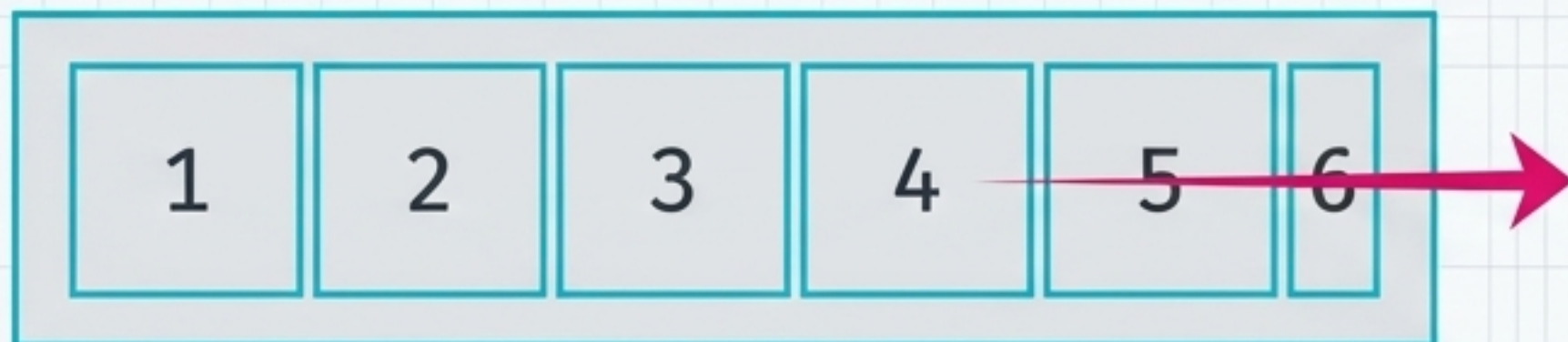
# FLEXBOX: WRAP & FLOW

A Detailed Study Guide to Multi-Line Layouts





# THE PROBLEM: WHEN FLEX ITEMS DON'T BEND, THEY BREAK



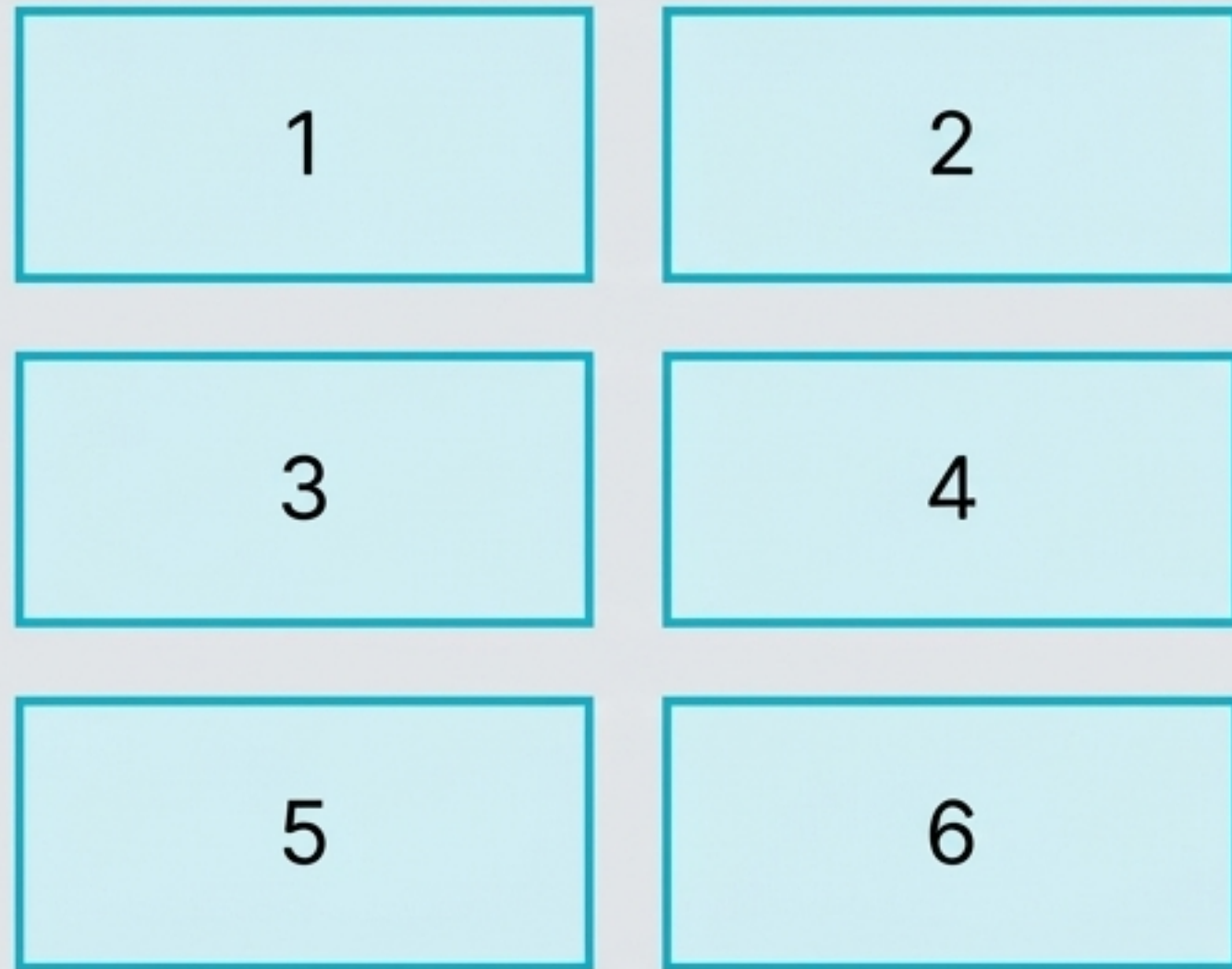
By default, a flex container forces all its children onto one line. This leads to overflow, unwanted scrollbars, or unreadably small items on smaller screens.

This is the difference between “crammed” and “comfortable.”

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



# THE SOLUTION: LAYOUTS THAT ADAPT, NOT BREAK



Wrapping allows items to flow onto new lines when they run out of space on the main axis. This creates the fluid, responsive layouts that are essential for modern web design. The source order stays the same.

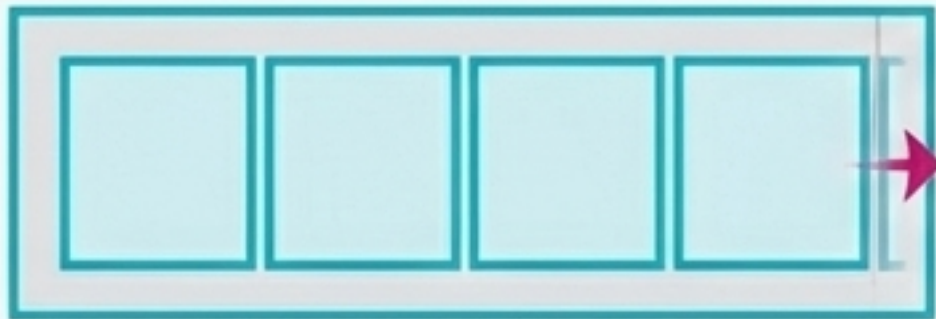
```
.container {  
  display: flex;  
  flex-wrap: wrap; /* The solution */  
}
```



# THE CORE CONTROL: `flex-wrap`

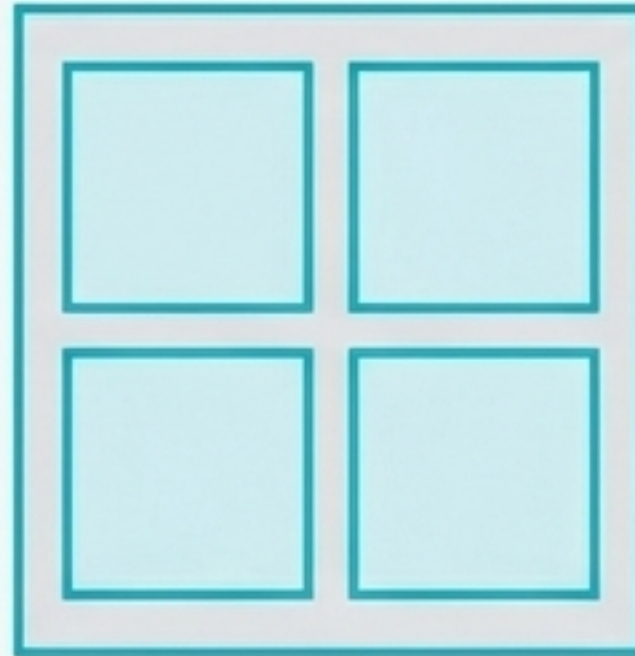
## `nowrap` (Default)

Items are forced onto a single line. They will shrink to fit or overflow the container.



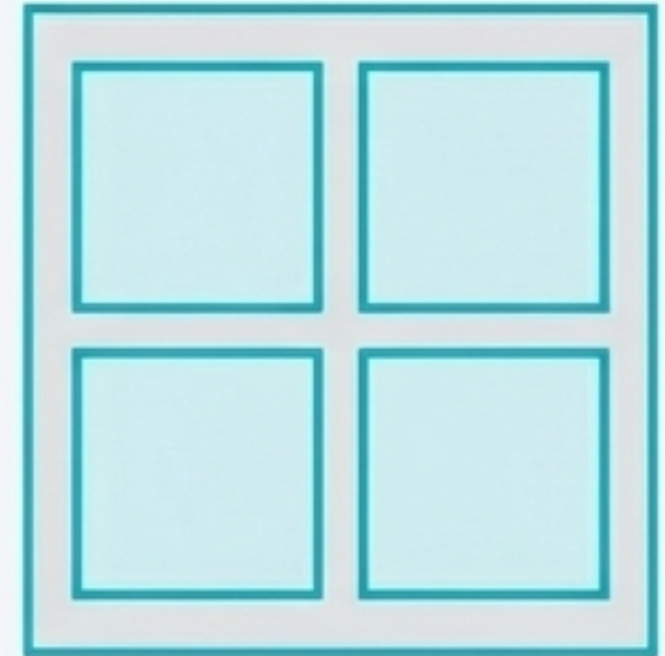
## `wrap`

Items flow onto new lines in the cross-axis direction. For a `row`, new lines appear below.



## `wrap-reverse`

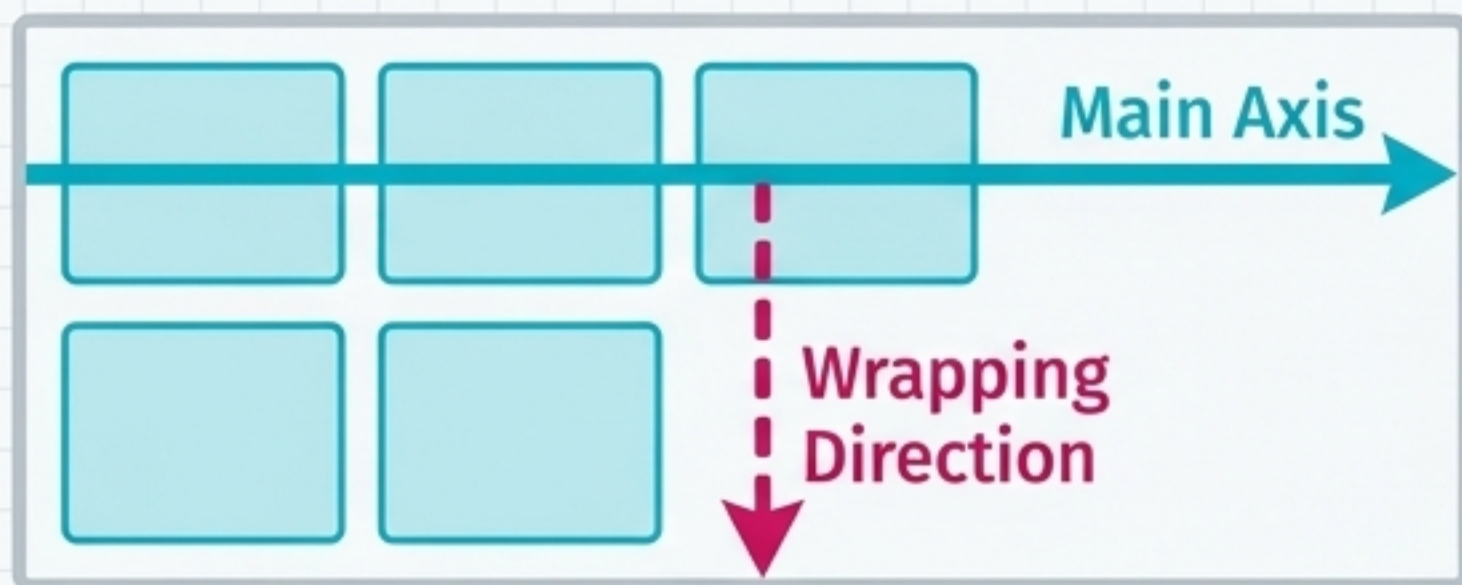
Items flow onto new lines in the opposite direction. For a `row`, new lines appear above.





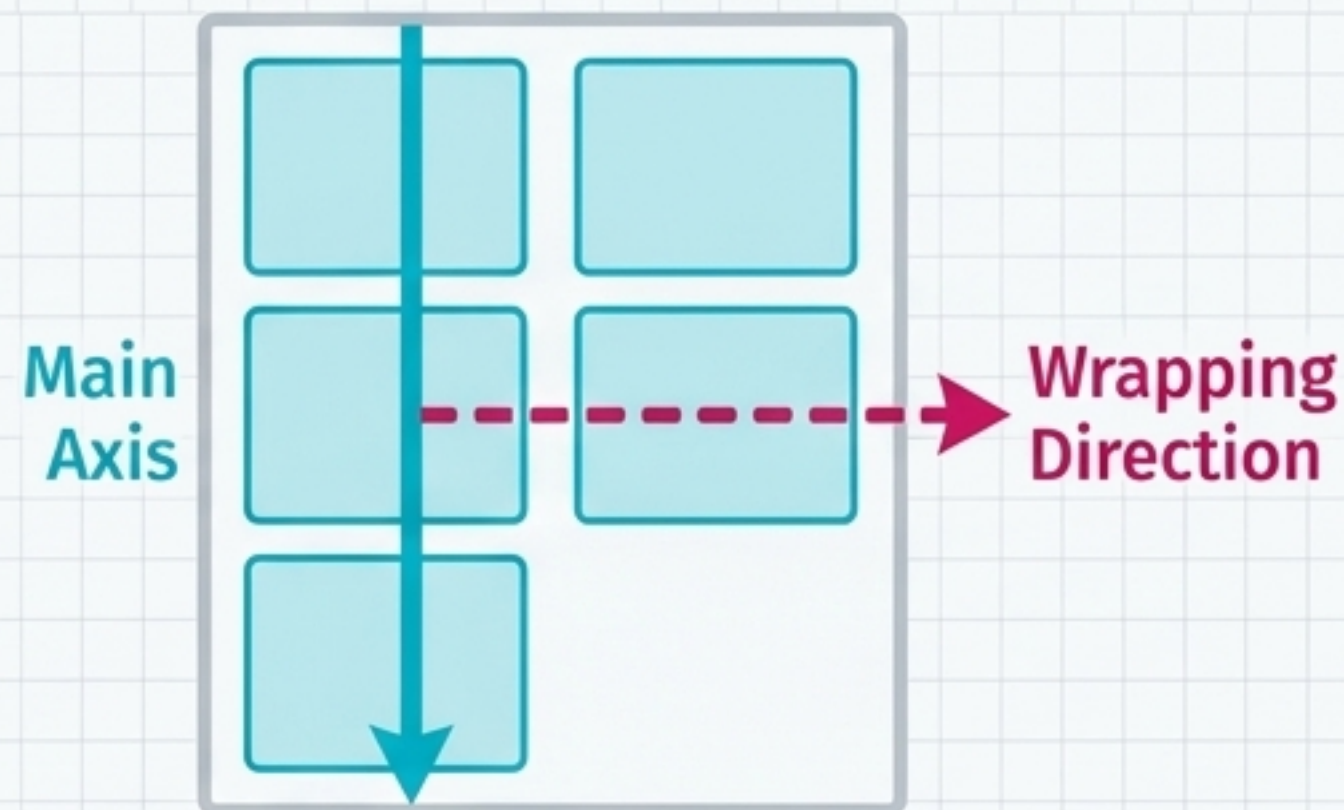
# WRAPPING DIRECTION DEPENDS ON THE MAIN AXIS

``flex-direction: row``



With `flex-direction: row`, wrapping creates new **rows** stacked along the cross-axis (top to bottom).

``flex-direction: column``



With `flex-direction: column`, wrapping creates new **columns** beside the original line (left to right).



# THE PRO'S SHORTHAND: `flex-flow`

Combining Direction and Wrapping in a Single Declaration

```
.container {  
  display: flex;  
  flex-direction: row;  
  flex-wrap: wrap;  
}
```



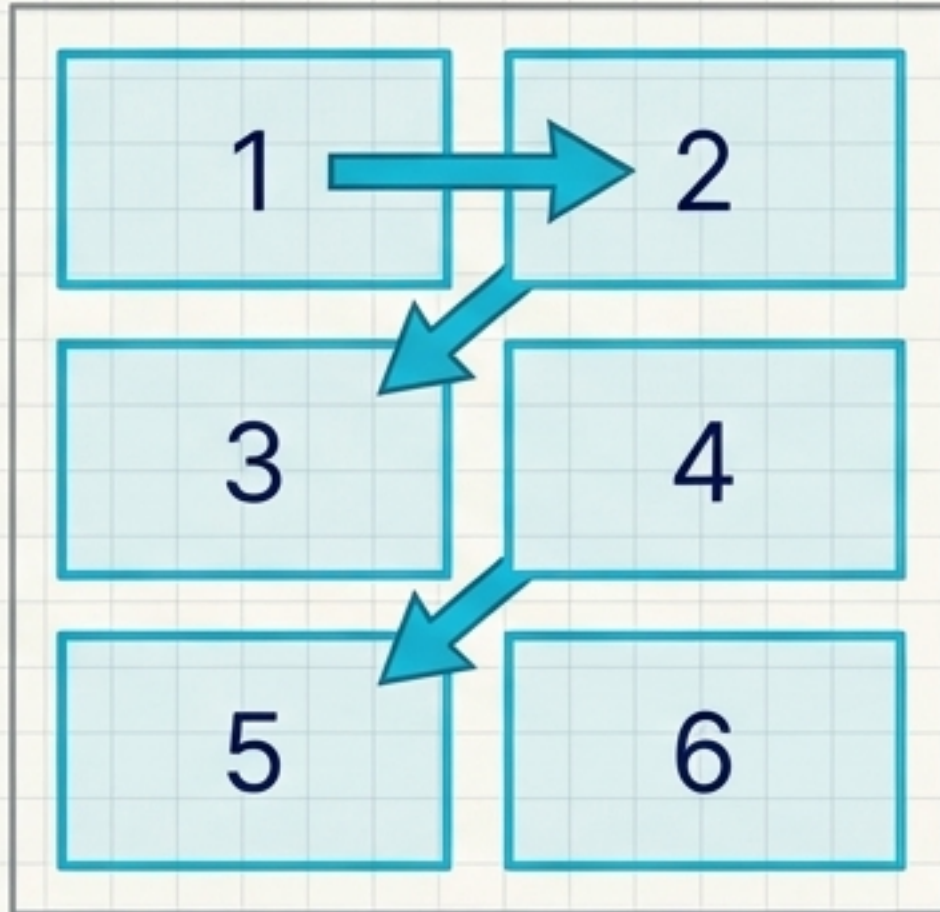
```
.container {  
  display: flex;  
  flex-flow: row wrap;  
}
```

The `flex-flow` property is a convenient shorthand for `flex-direction` and `flex-wrap`. This is the combo move you will most often see and use in real projects.



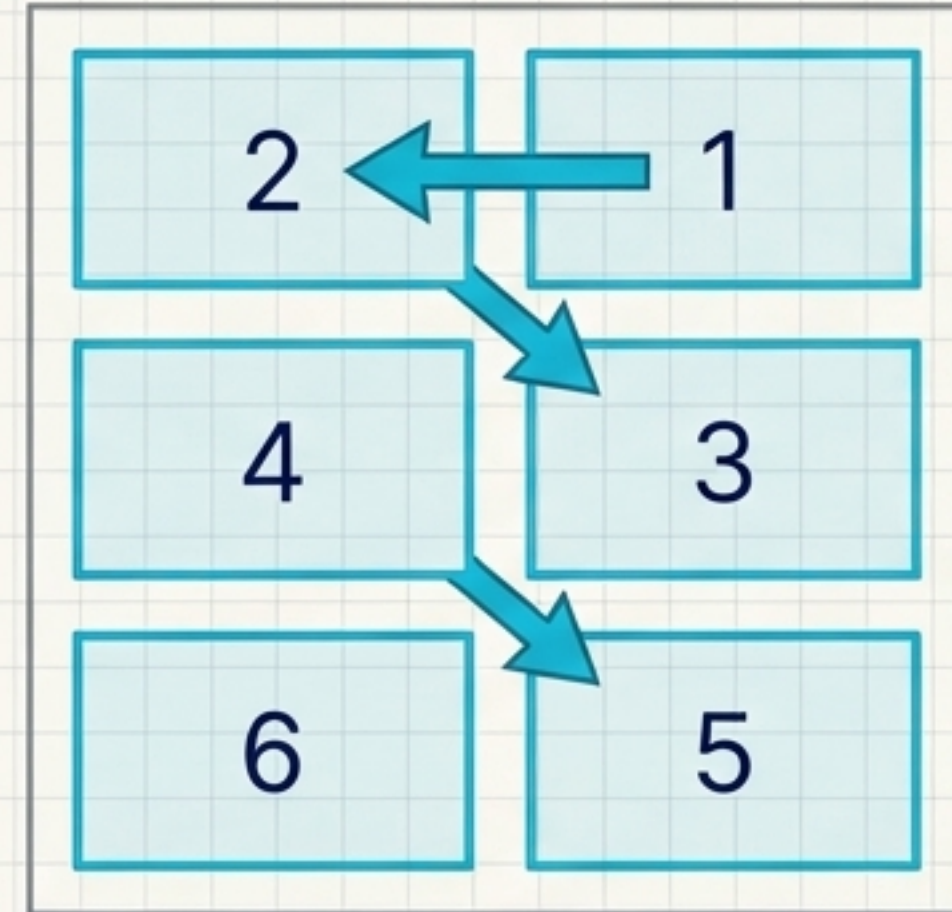
# PRACTICAL `flex-flow` COMBINATIONS

**flex-flow: row wrap;**



Ideal for responsive card galleries, product listings, and tag clouds.

**flex-flow: row-reverse wrap;**

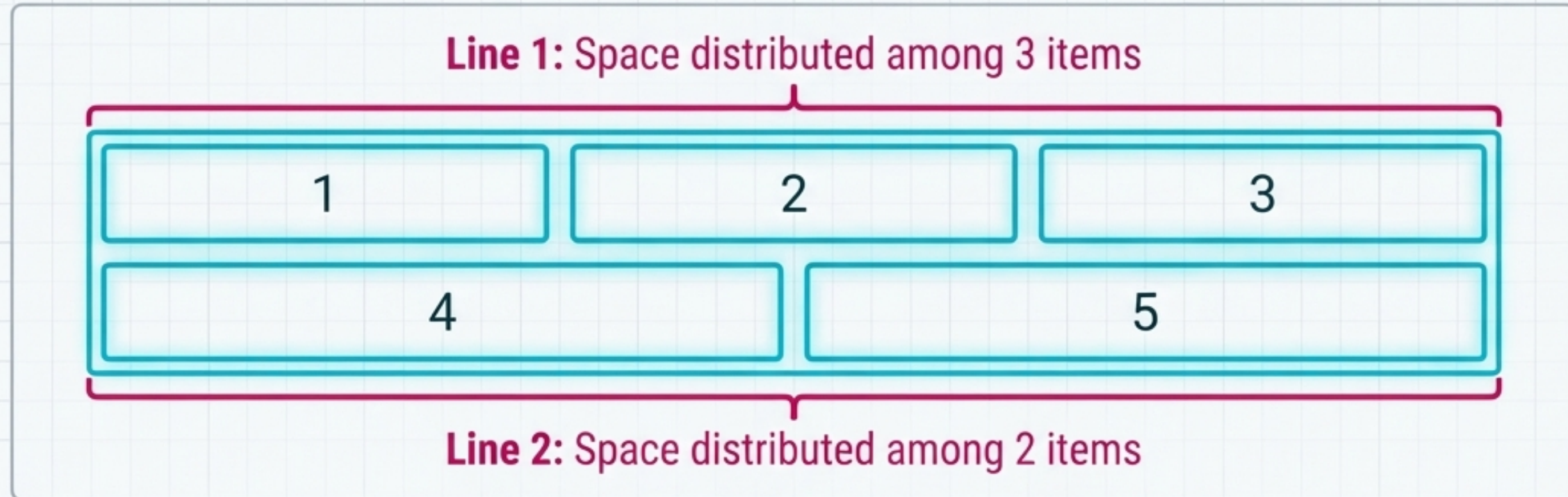


Useful for layouts aligned to the right edge or for right-to-left language interfaces.



# WRAPPING AND SIZING WORK TOGETHER

Sizing properties (**flex-grow**, **flex-shrink**, **flex-basis**) are calculated per line.



- flex-grow distributes extra space *within a single line*.
- flex-shrink determines how items yield space *on a single line*.
- flex-basis sets the initial size before wrapping is considered.



# DEFINING THE BOUNDARIES OF `flex-wrap`



## WHAT `wrap` DOES

- Decides if items can form new lines.
- Sets the stacking direction of new lines (along the cross-axis).
- Creates independent “flex lines,” each with its own space negotiation.

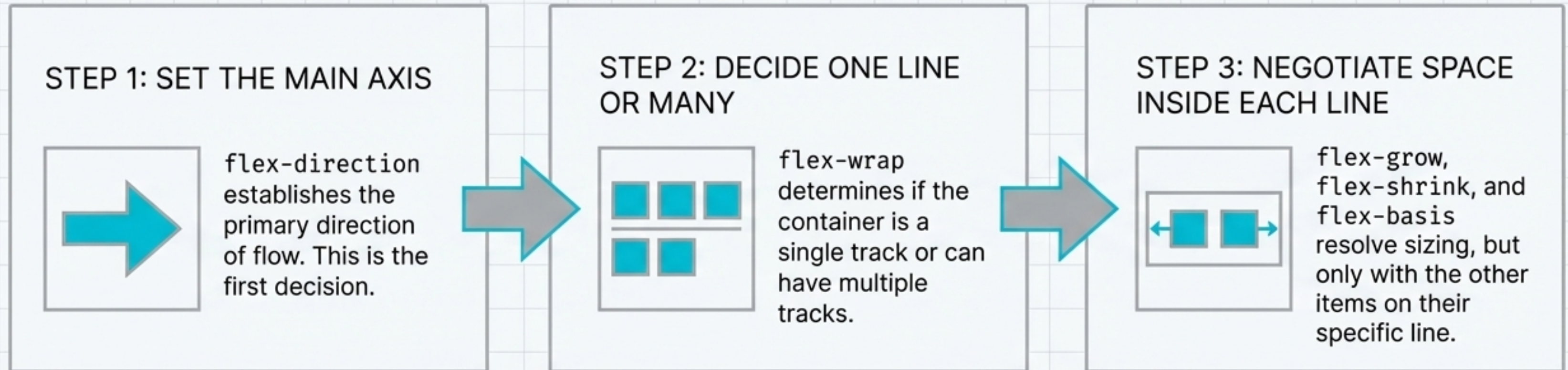


## WHAT `wrap` DOES NOT

- Control spacing *between* lines (use `row-gap` or `gap` for that).
- Align items across different lines (use `align-content` for that).
- Change the HTML source order of the items.



# THE MENTAL MODEL FOR MASTERY: A 3-STEP PROCESS





# AVOID THESE COMMON WRAPPING PITFALLS

## Pitfall 1: Forgetting the Default

**Issue:** “My items are overflowing! Why aren’t they wrapping?”

**Solution:** Remember, `flex-wrap: nowrap;` is the default. You must explicitly set `flex-wrap: wrap;` or `flex-flow: row wrap;`.

## Pitfall 2: Confusing Wrap with Grid

**Issue:** “Why don’t items in the second row align with items in the first?”

**Solution:** Flexbox lines are independent. Each line calculates its own spacing. For a strict two-dimensional grid, [CSS Grid](#) is the appropriate tool.

## Pitfall 3: Expecting Wrap to Add Space

**Issue:** “My wrapped rows are touching each other.”

**Solution:** `flex-wrap` only creates the lines; it doesn’t add space between them. Use the [gap](#) or [row-gap](#) property for that.



# KEY TAKEAWAYS: WRAP & FLOW

- Wrapping solves overflow by allowing items to flow onto new lines.
- flex-wrap is the core property with values wrap, nowrap, and wrap-reverse.
- flex-flow is the essential shorthand for combining flex-direction and flex-wrap.
- Sizing and space distribution are calculated independently for each line.



## WHAT'S NEXT?

Now that you can create multiple lines, the next step is to control their alignment and spacing. We turn our attention to **align-content** and **gap** next—spacing that finally behaves like it always should have.





Continue your learning at Professor Solo / WebDevTnT.