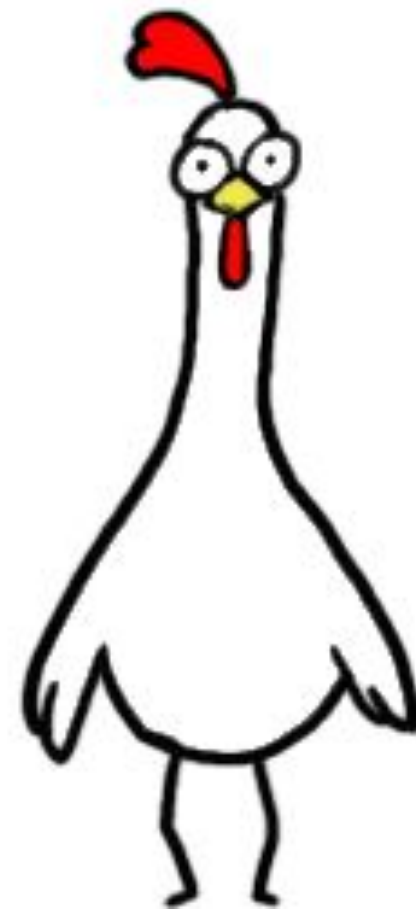# FLEXBOX

What is CSS Flexbox?

◉ The Flexbox Layout module aims at providing a more efficient way to lay out, align, and distribute space among items in a container, even when their size is unknown and/or dynamic (thus the word "flex").

◉ Again but with less jargon: An efficient way to have our parent elements communicate to our children elements to behave in a certain way even if we don't know much about our children

Can I use it?

caniuse.com

IE	Edge *	Firefox	Chrome	Safari	Opera	iOS Safari *	Opera Mini *	Android Browser *	Chrome for Android
8			45					4.3	
9			46					4.4	
10		43	47			8.4		4.4.4	
11	13	44	48	9	34	9.2	8	47	47
	14	45	49	9.1	35	9.3			
		46	50		36				
		47	51						

FULLSTACK

REMINDER: BLOCK ELEMENTS & INLINE ELEMENTS

- **A block level element always starts on a new line and takes up the full width available**

  - Ex: divs, paragraphs, forms, tables, lists

- **An inline level element does not start on a new line and only takes up as much width as necessary**

  - Ex: spans, emphasis, inputs

- **[Documentation]**

- **We will be focusing on block level in this lecture's content**

The Container
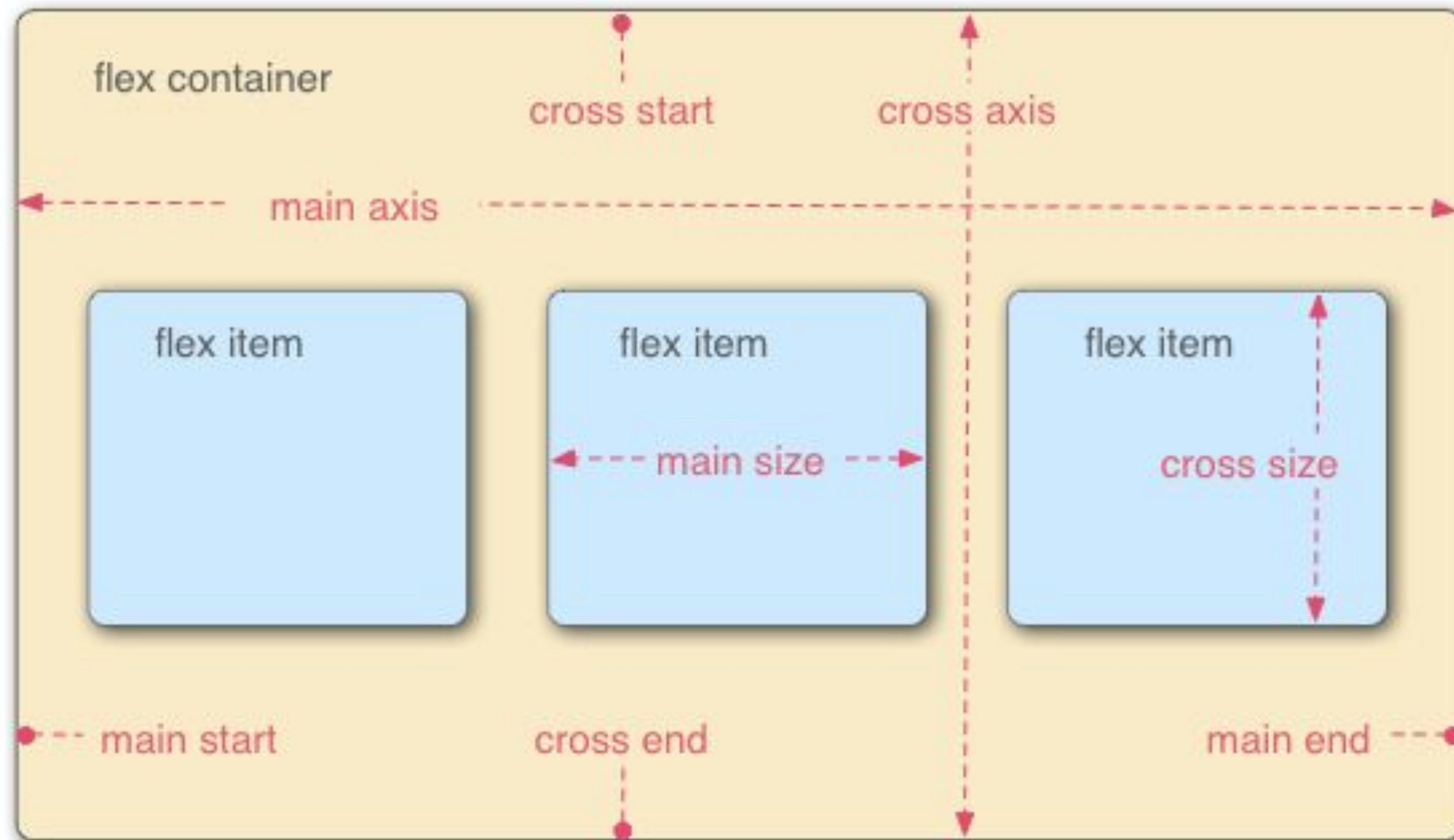
Flexbox Container: display

- display: flex defines an element as a flex container and enables a flex context for all its direct children

```
.container {
  display: flex; /* or inline-flex */
}
```
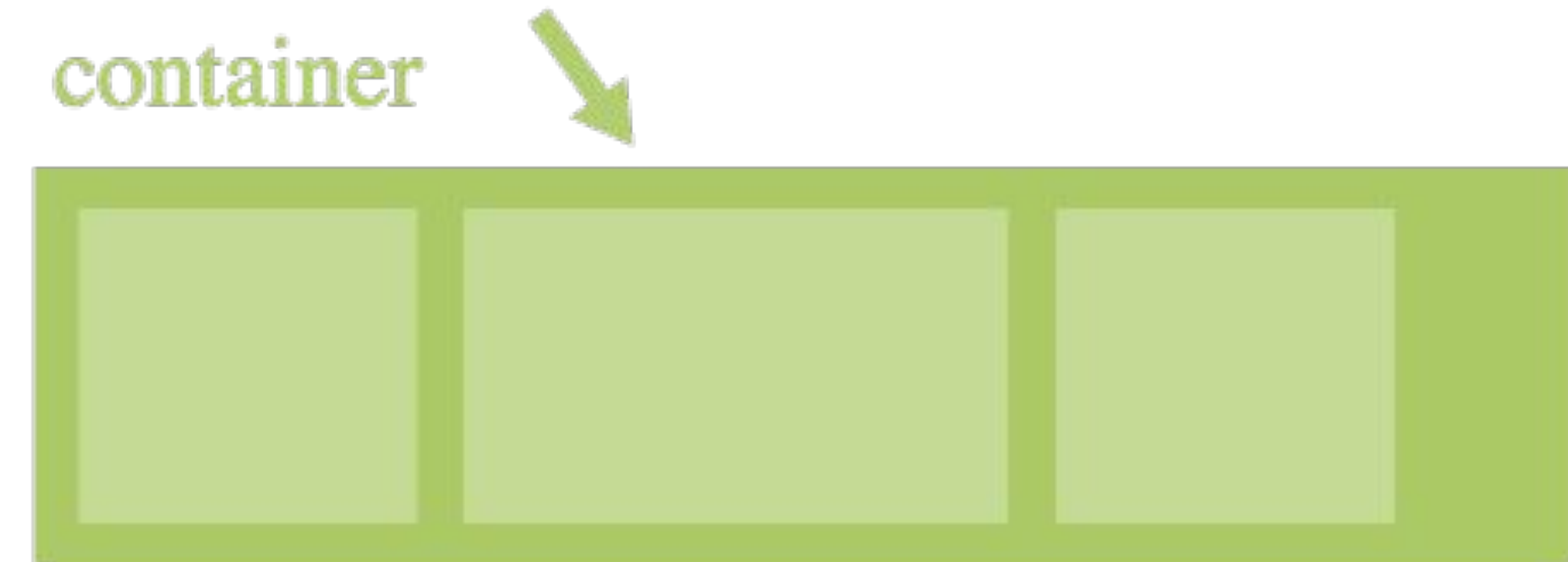
Callback to Block Elements

- What we are saying is something like display: block flex

  - The "outer display" defines a block level element (or inline)

  - The "inner display" defines the flex layout

```
.container {
  display: flex; /* or inline-flex */
}
```
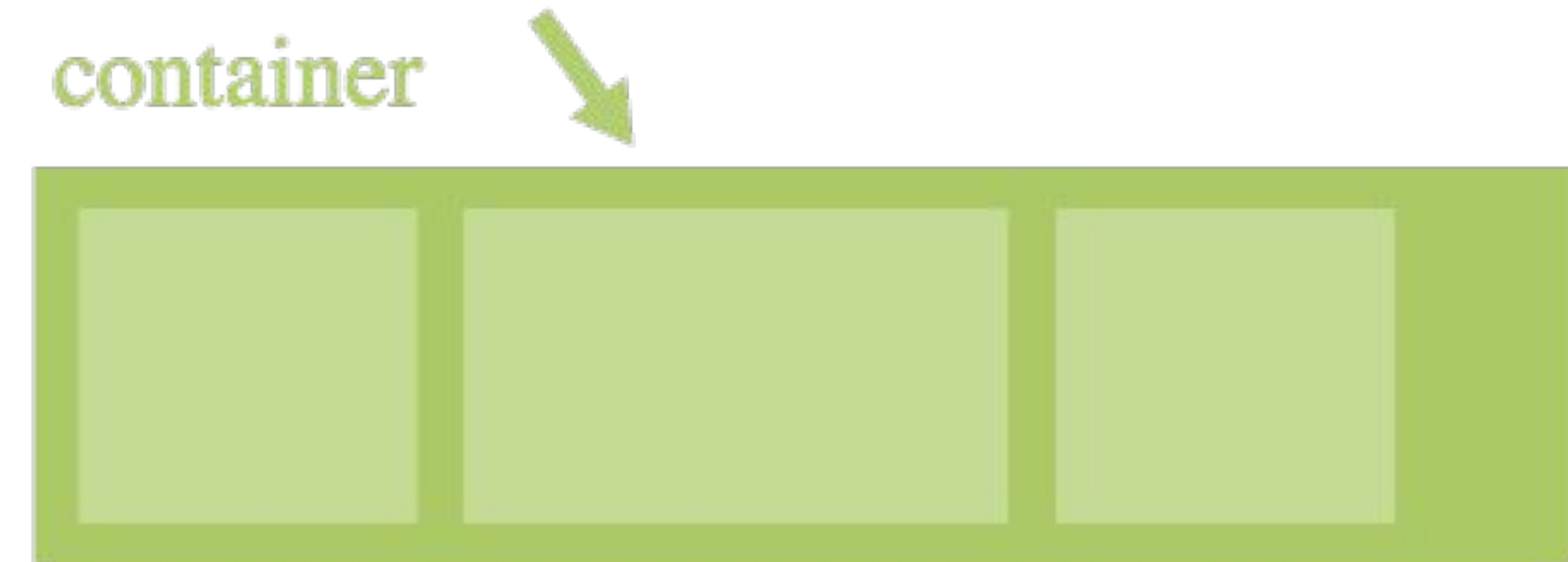
Flexbox Container

- Flexbox gives the container the ability to alter its items' dimensions (and order) to best fill the available space.

Flexbox Container

⊙ A flex container expands flexible items to fill free space, or shrinks them to prevent overflow.
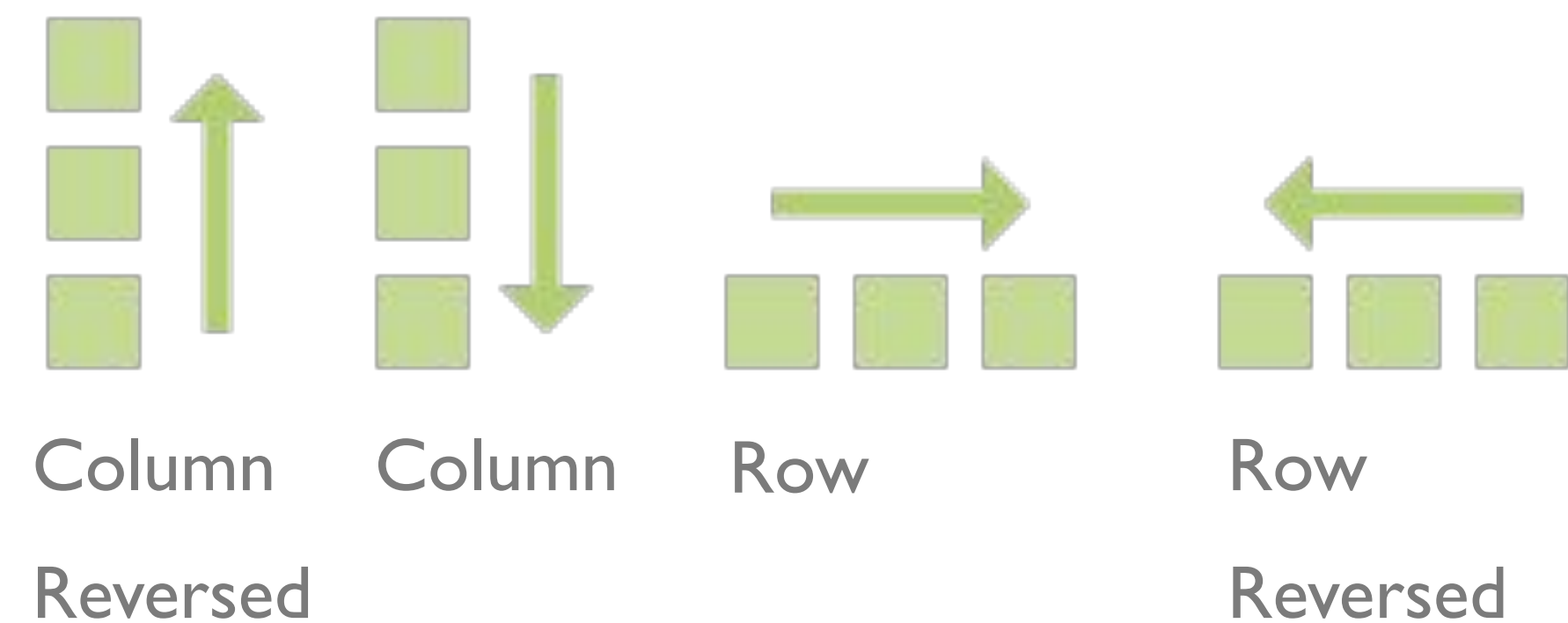

container

Flexbox Container: flex-direction

- Flexbox is (aside from optional wrapping) a single-direction layout concept. Think of flex items as primarily laying out either in horizontal rows or vertical columns.

- "main axis" and "cross axis"

FULLSTACK

Flexbox Container: Direction

◉ Think of flex items as primarily laying out either in horizontal rows or vertical columns.

Column   Column   Row   Row

Reversed   Reversed

```
.container {
  flex-direction: row | row-reverse | column | column-reverse;
}
```
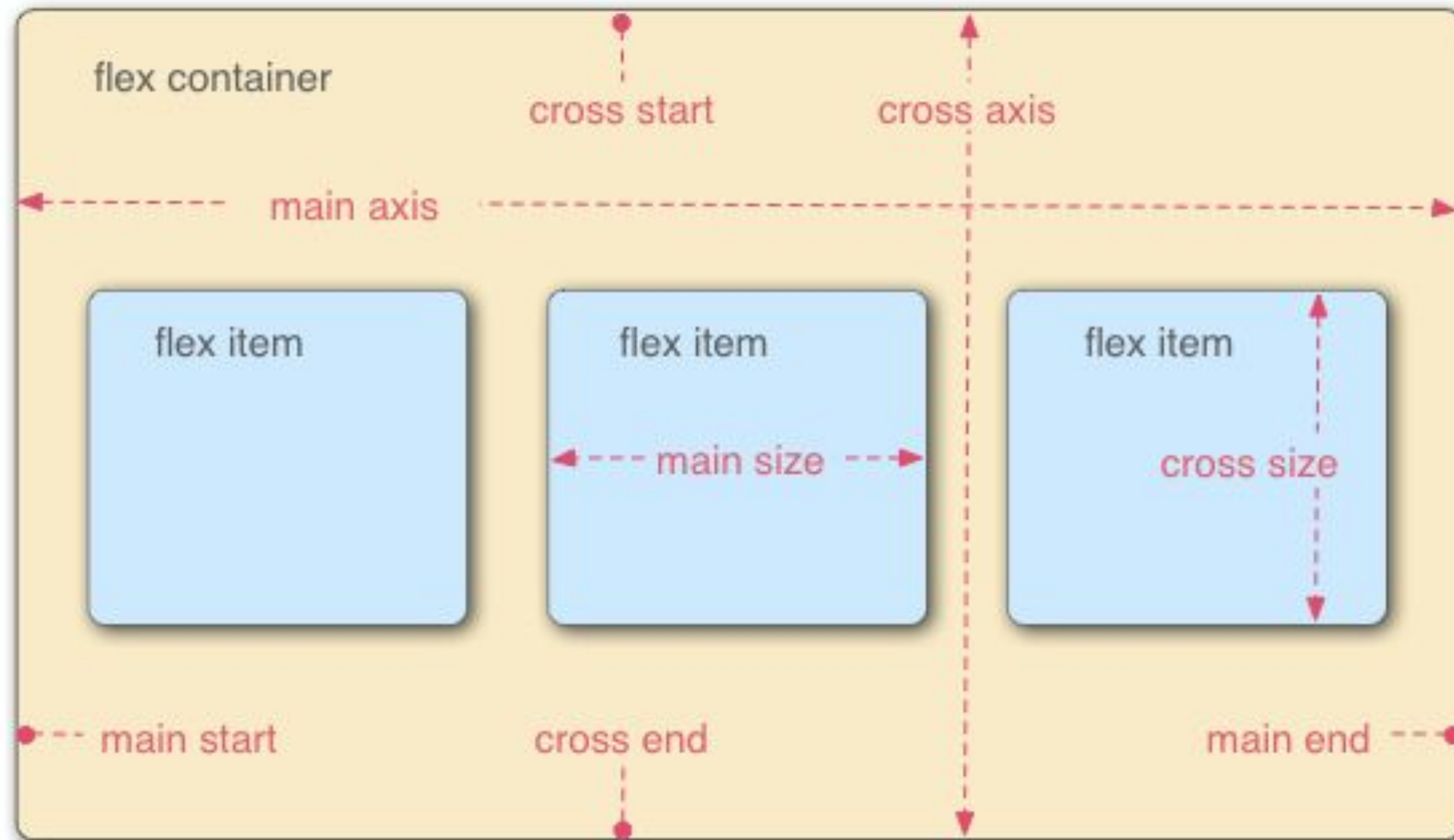
Flexbox Container: wrap

- Items will all try to fit onto one line. Items can wrap as needed with this property. Direction also plays a role here, determining the direction new lines are stacked in.
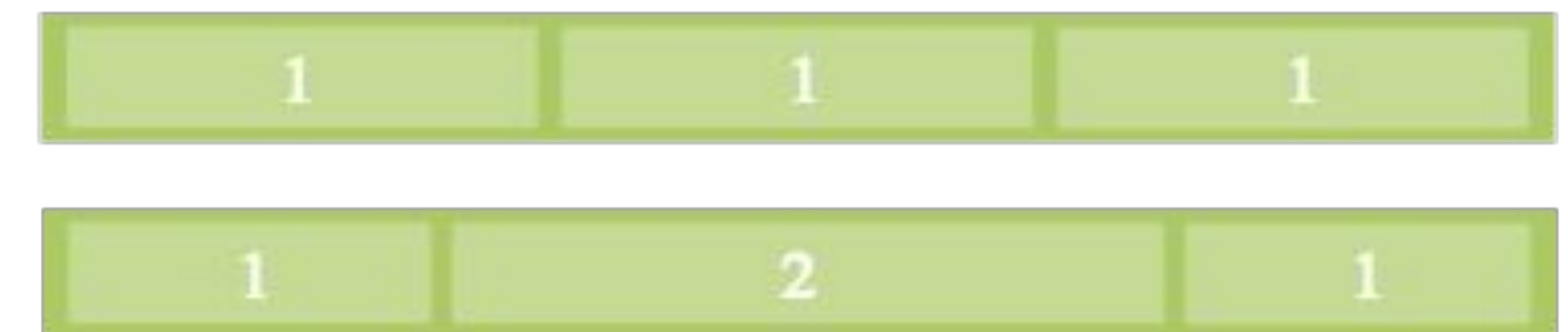


```
.container{
    flex-wrap: nowrap | wrap | wrap-reverse;
}
```
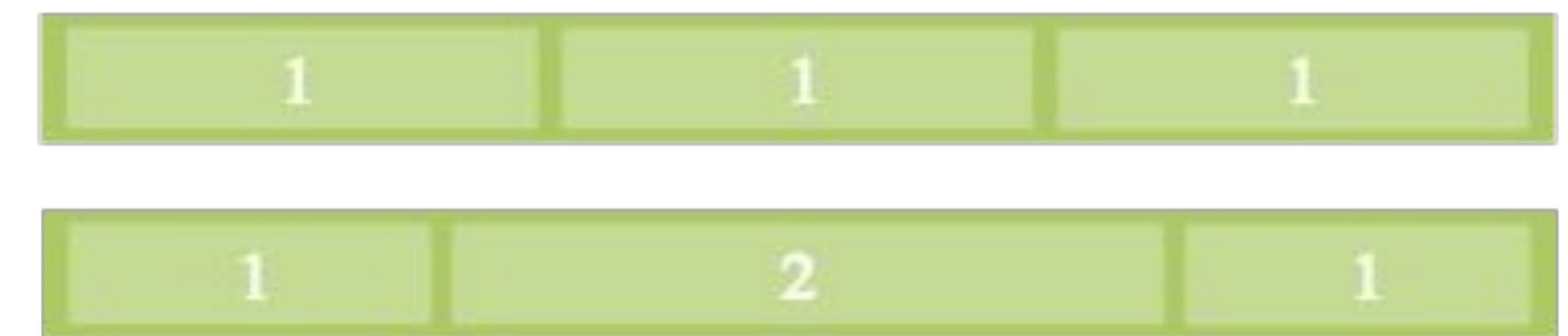
The Items

Flexbox Items: flex-grow

⊙ Flex items can grow if necessary. This property accepts a unit-less value that serves as a proportion. It dictates what amount of the available space inside the flex container the item should take up.


```
.item {
  flex-grow: <number>; /* default 0 */
}
```

Flexbox Items: flex-grow

⊙ eg. if all items have flex-grow set to 1, every child will set to an equal size inside the container. If set to 2, that child would take up twice as much space as the others.


```
.item {
  flex-grow: <number>; /* default 0 */
}
```

FULLSTACK

Flexbox Items: flex-shrink

◉ This defines the ability for a flex item to shrink if necessary. Negative numbers are invalid.

◉ Not that necessary

```
.item {
  flex-shrink: <number>; /* default 1 */
}
```

Flexbox Items: flex-basis

◉ Like width property (or height, depending on flex-direction).

◉ If a relative value, indicates proportion of that item's width that should be applied.

◉ Default size of element before flex-grow or flex-shrink kick in

```
.item {
  flex-basis: <length> | auto; /* default auto */
}
```

Flexbox Items: flex

⊙ This is the shorthand for flex-grow, flex-shrink and flex-basis combined. The second and third parameters (flex-shrink and flex-basis) are optional. Default is 0 1 auto.

```
.item {
  flex: none | [ <'flex-grow'> <'flex-shrink'>? || <'flex-basis'> ]
}
```

WORKSHOP