FLEXBOX AND GRID LAYOUT

Diego Eis

I love work with web.
And I lost 50 pounds. ;-)

@diegoeis
@tableless

http://**tableless.com.br**
http://**medium.com/@diegoeis**
http://**slideshare.net/diegoeis**

SEVENTH-DAY
ADVENTIST®
CHURCH

http://bit.ly / livro-como-ser-frontend


LOCAWEB

LOCAWEB

OPEN SOURCE

opensource.locaweb.com.br

TABLELESS

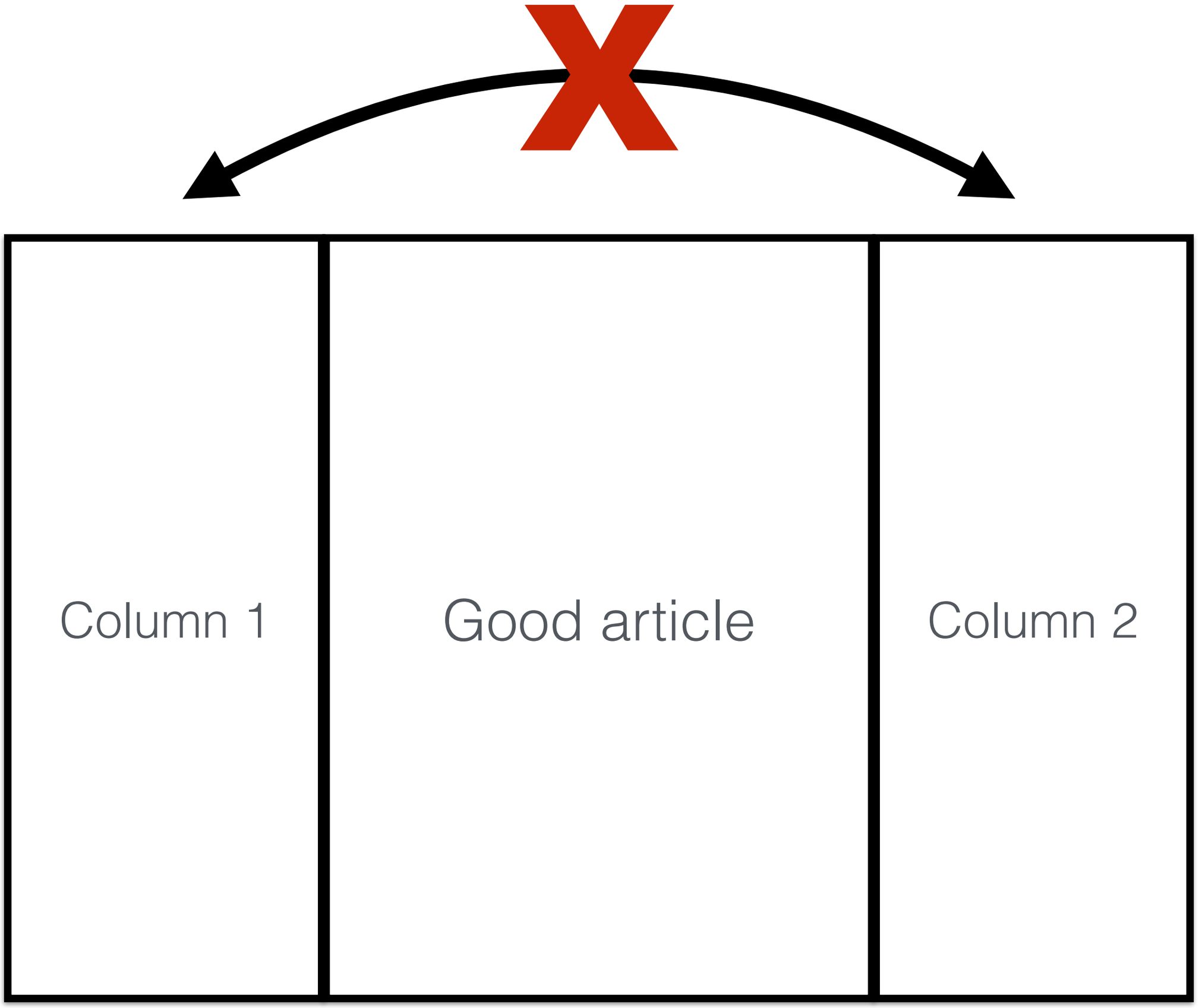
Gambi.

Until now we only used Macgyver ways to structure layouts.

Table

Tables made history. They changed the way how we show and structure content in the websites, **but...**

WRONG!


X
Column 1
Good article
Column 2

No semantic.

Float

Float give us some flexibility, **but...**

Float affect other elements

Forcing you to use other properties and techniques to solve some problems: clearfix, overflow, faux columns, double margins (old ie) etc etc etc…

Float depends of HTML structure

You need to put the HTML elements in right place and order to make this right.

How to solve the problem of structuring layouts?

Three solutions to different problems

- **Grid Layout** to structure parent elements.
- **Flexbox** to control the struture of childs.
- **Template Layout** to control the flow of content.

Template Layout

At the moment, it defines a *typographic grid* for CSS. It has features to set up a grid-based template, to style the *slots* of the template and to flow content into them.

Today, let's talk about Grid Layout and Flexbox

Grid Layout

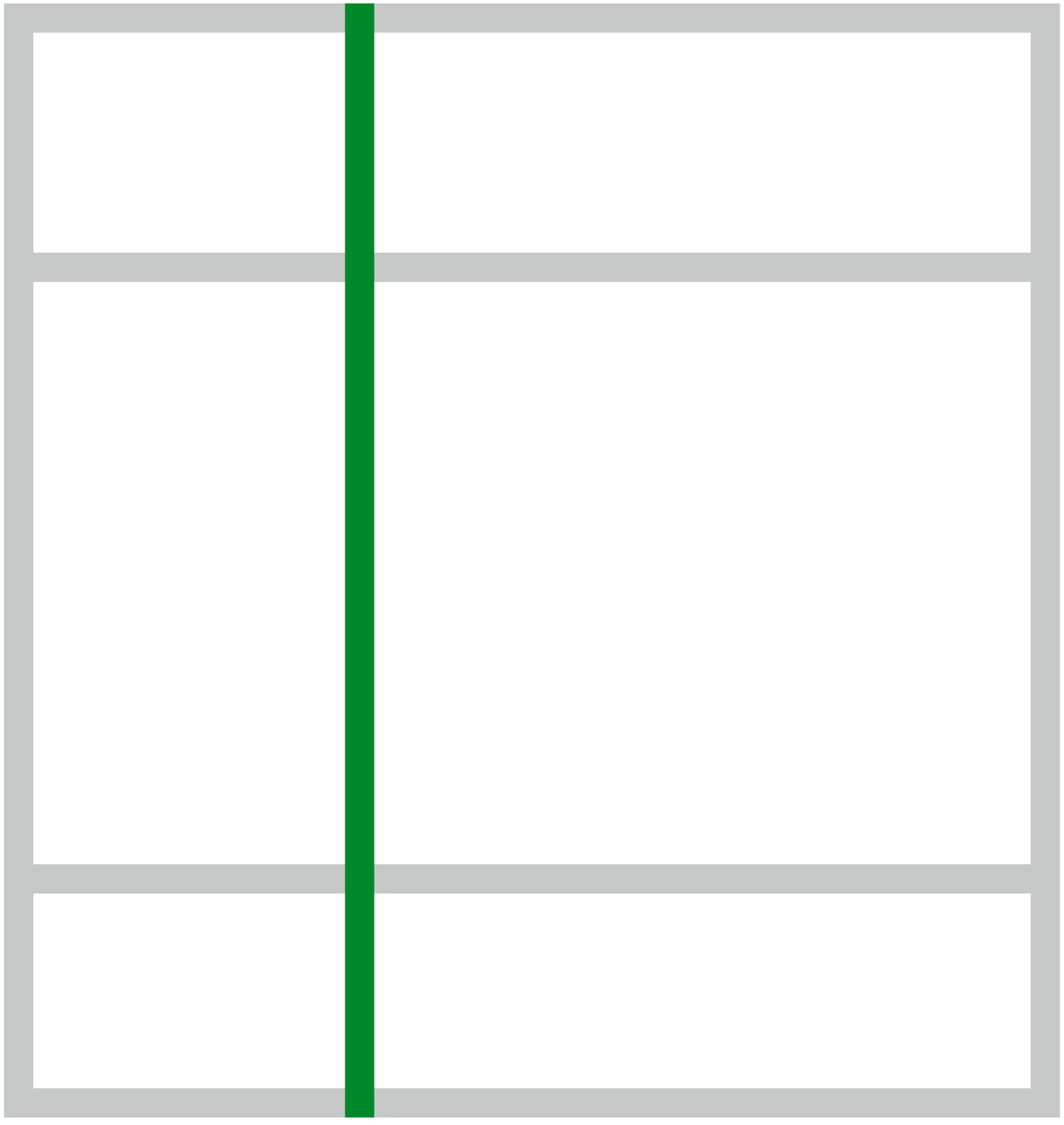
This CSS module defines a two-dimensional grid-based layout system, optimised for user interface design. In the grid layout model, the children of a grid container can be positioned into arbitrary slots in a flexible or fixed predefined layout grid.

Grid terminology

Grid container

A grid container establishes a new grid formatting context for its contents.

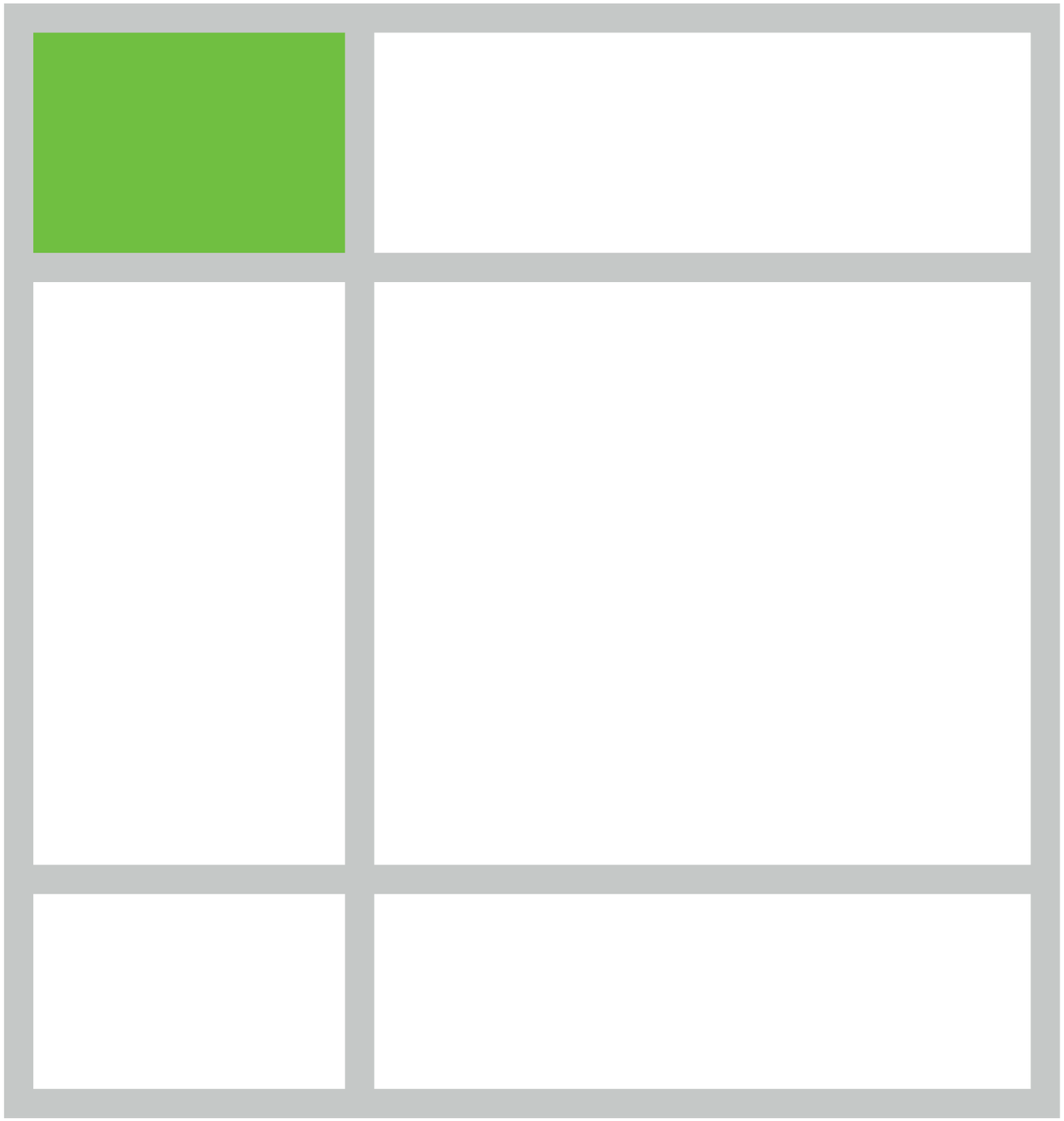
Grid lines



Grid lines are horizontal or vertical lines between grid cells. They can be named or referred by numbers.

The highlighted line in the image is the column line 2.

Grid cell



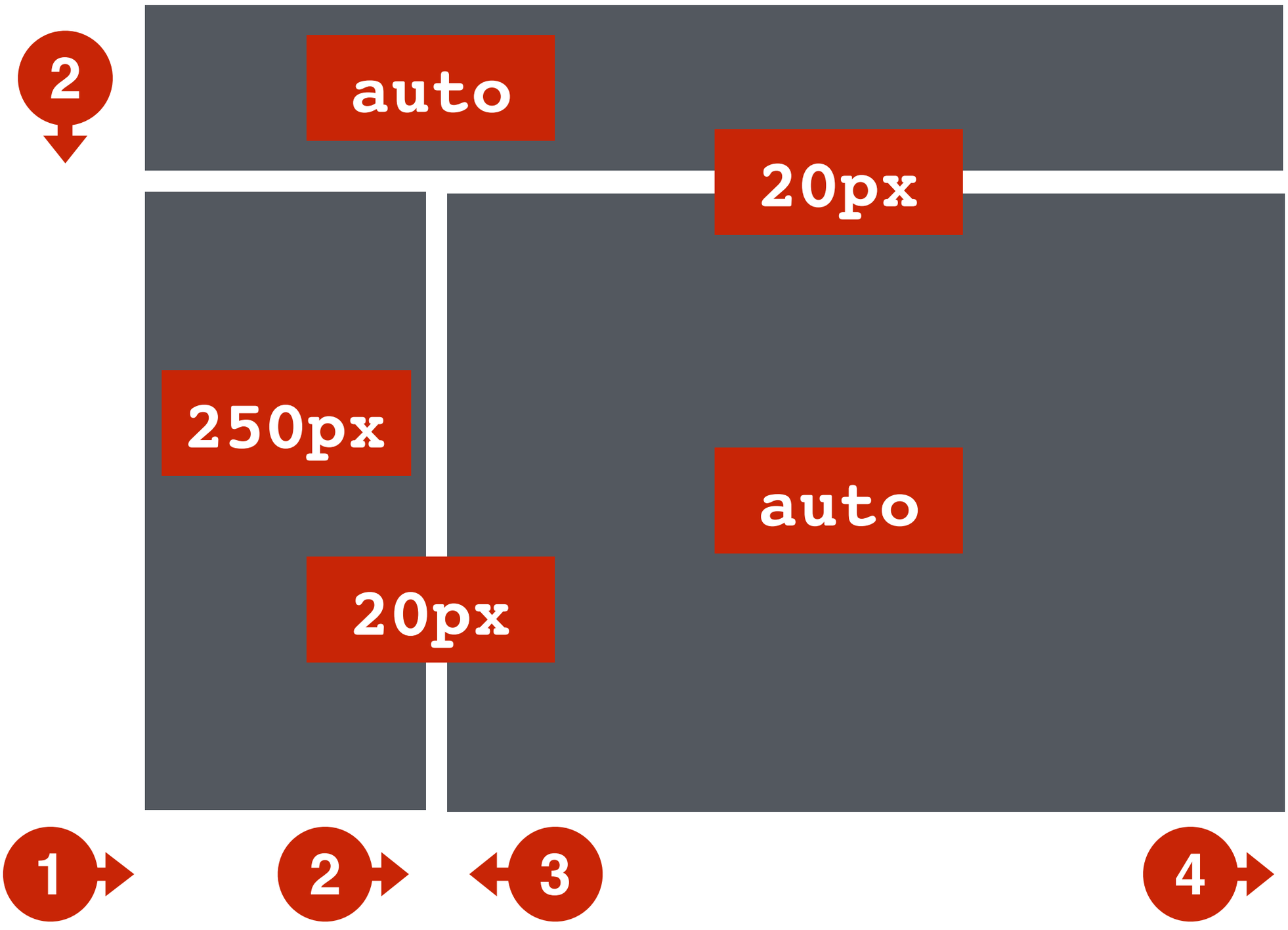
It is the **space between two adjacent row and two adjacent column grid lines**. It is the smallest unit of the grid that can be referenced when positioning grid items

```
.main {

  // Enable the grid space
  display: grid;
  grid-template-rows: auto 20px auto;
  grid-template-columns: 250px 20px auto;

}
```

1

```
grid-template-columns: auto 250px 20px auto;
```

2



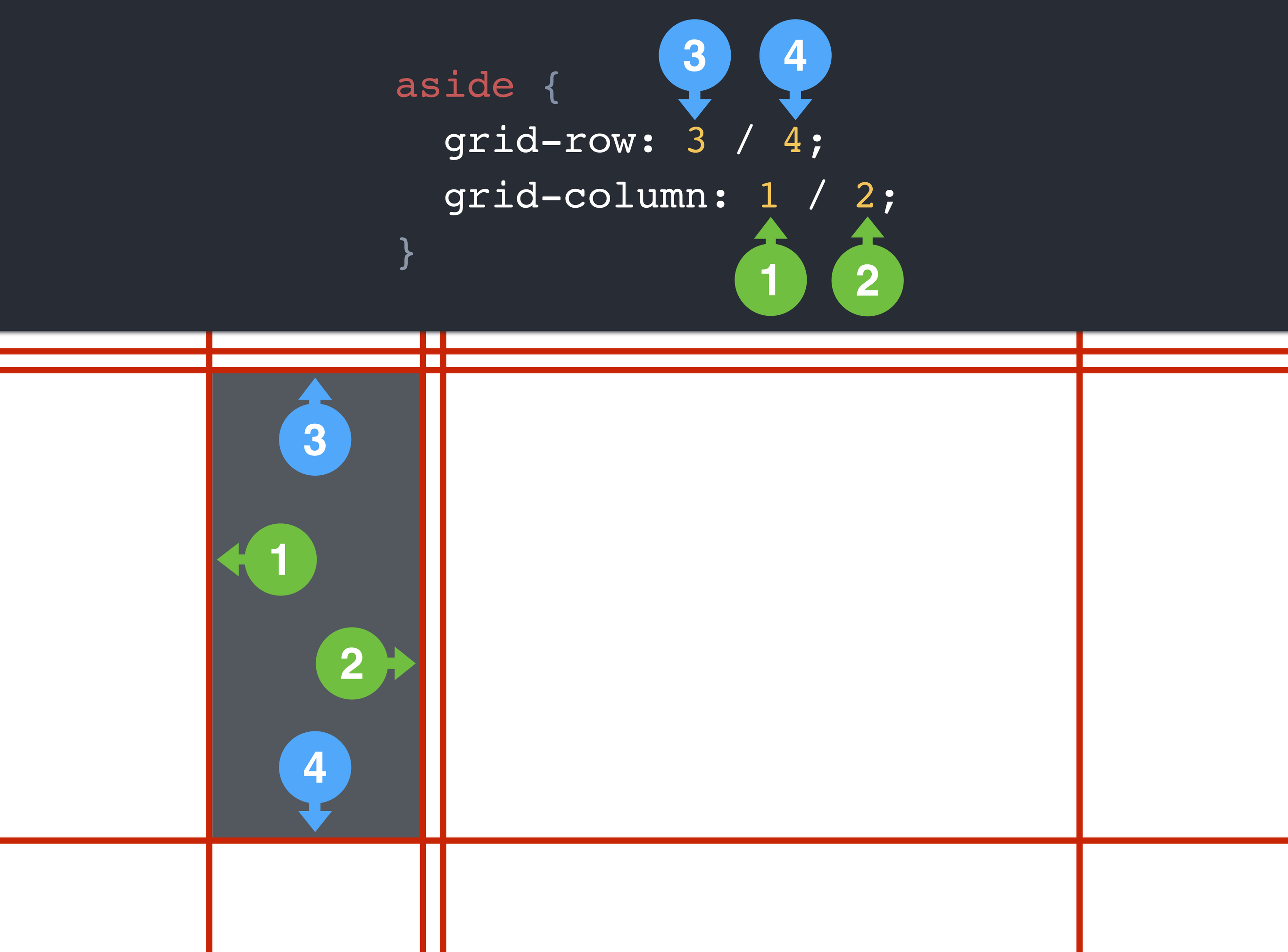
3

4

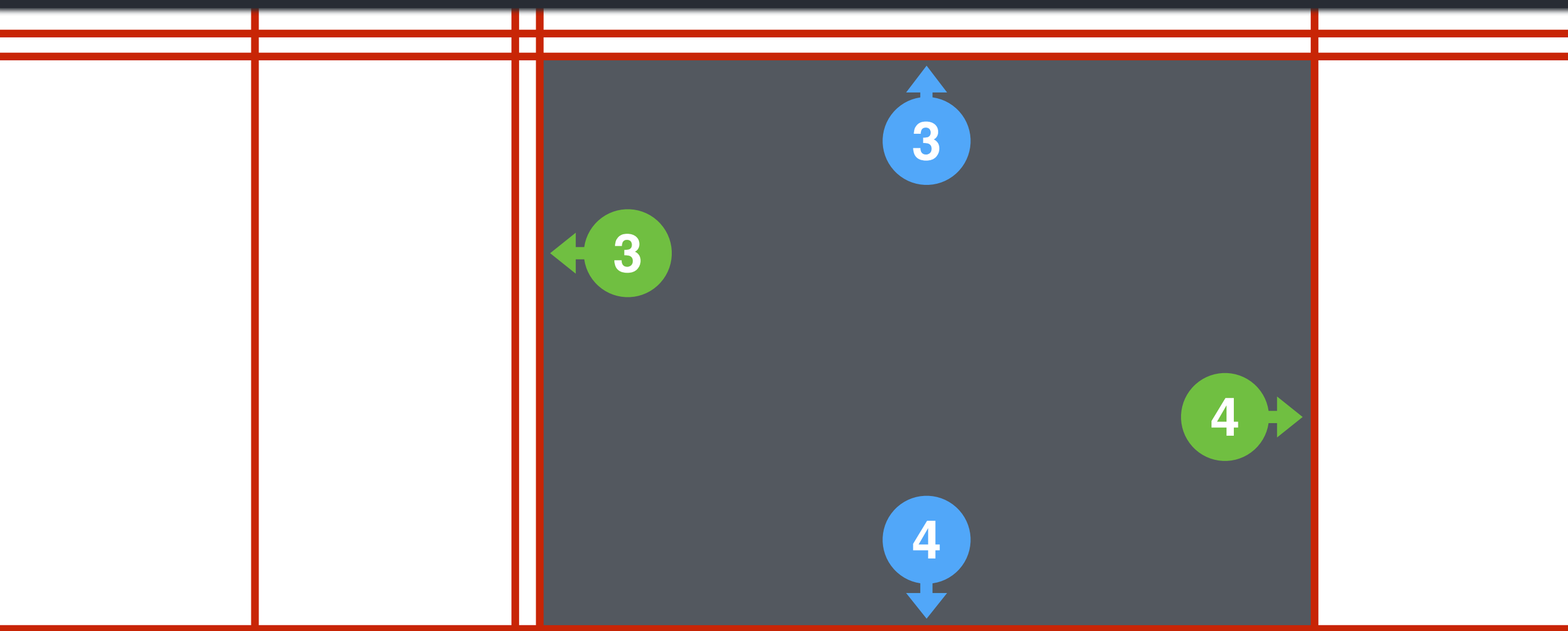
1 2 3 4


```
header {
  grid-row: 1 / 2;
  grid-column: 1 / 4;
}
```

```
aside {
  grid-row: 3 / 4;
  grid-column: 1 / 2;
}
```



```
.content {
  grid-row: 3 / 4;
  grid-column: 3 / 4;
}
```



header

sidebar

content

```
.main {

  display: grid;
  grid-template-rows: auto 20px auto;
  grid-template-columns: 250px 20px auto;

  grid-template-areas: “header header header"
                       “. . .”
                       “sidebar . article”

}
```

```
header {
  grid-area: header;
}

aside {
  grid-area: sidebar;
}

article {
  grid-area: article;
}
```

IE	Firefox	Chrome	Safari	
		1 31		
		1 36		
		1 37		
		1 39		
8		1 40		
9	31	1 41	7	
2 − 10	37	1 42	7.1	
2 − 11	38	1 43	8	1
2 − Edge	39	1 44	9	1
	40	1 45		1
	41	1 46		

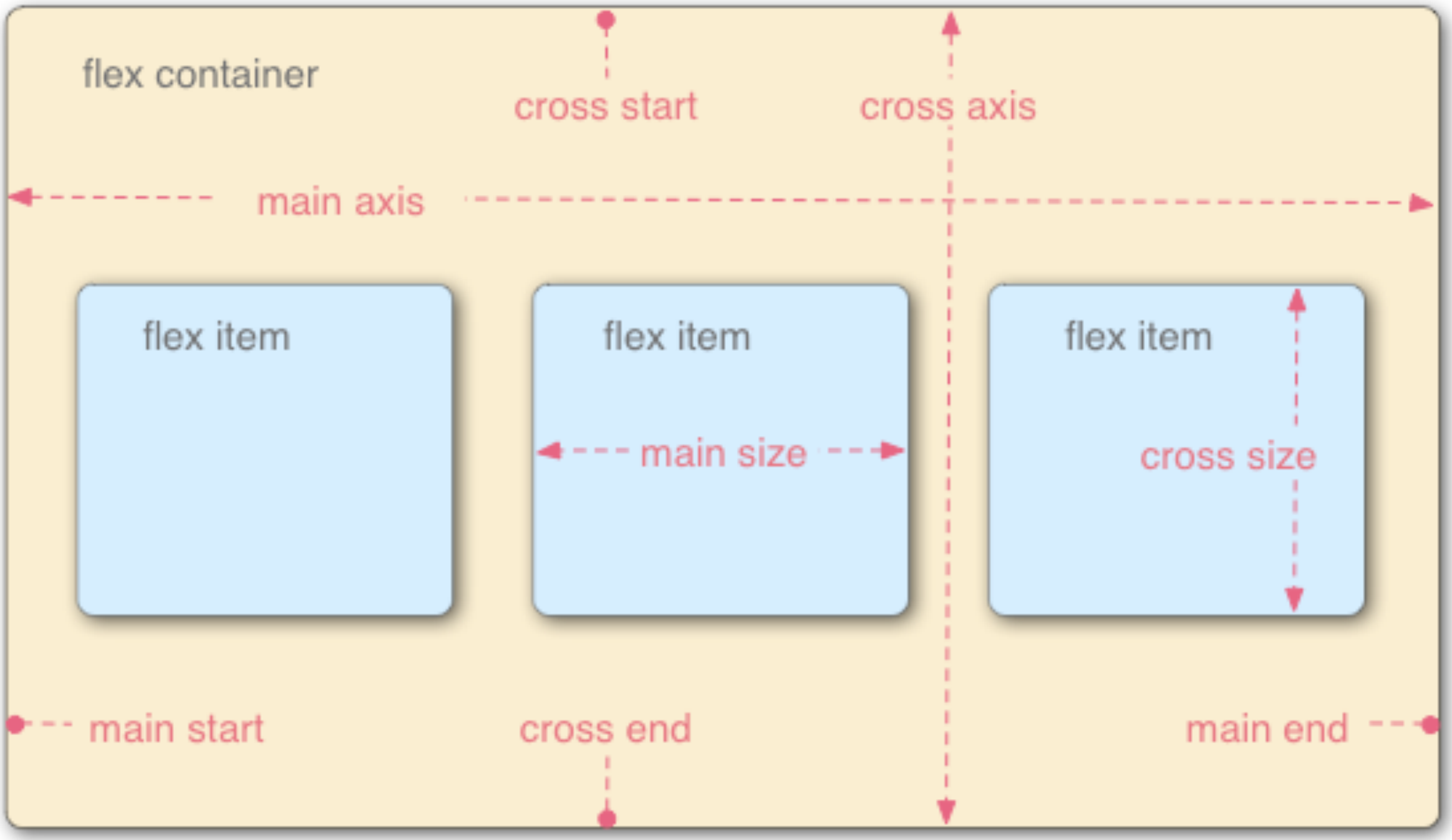
http://gridbyexample.com

Flexbox

Flexbox define how the childs will **fill the blank space available** of parent element.

Flex Container

First, we need to know the context where the flex items will work. The field where we will work is called Flex Container.



https://developer.mozilla.org/en-US/docs/Web/Guide/CSS/Flexible_boxes

flex-direction

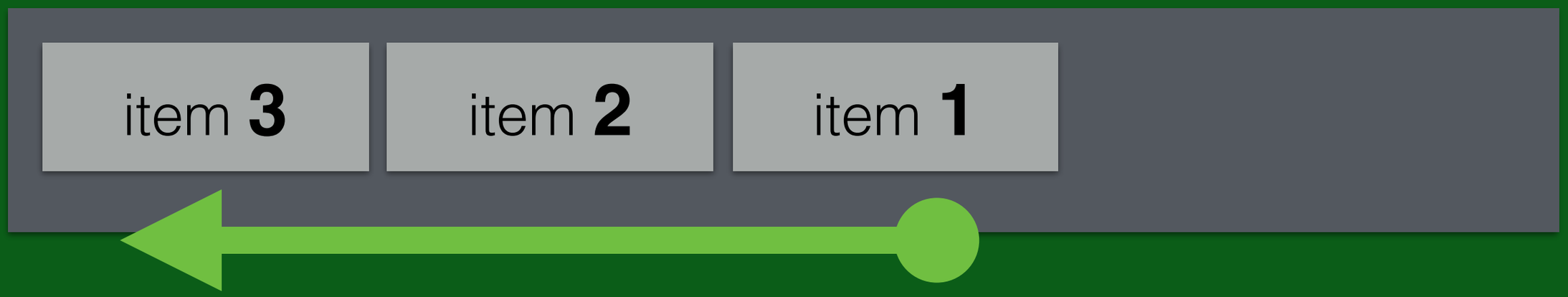
Define flow of the flex items placed in flex container.

flex-direction

row

default

row-reverse



flex-direction

column

item **1**

item **2**

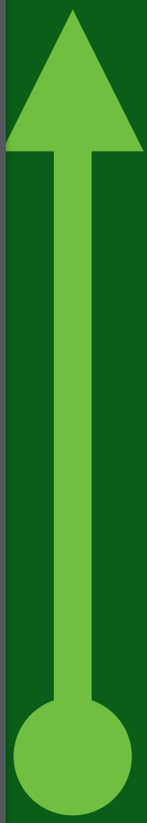
item **3**

column-reverse

item **3**

item **2**

item **1**

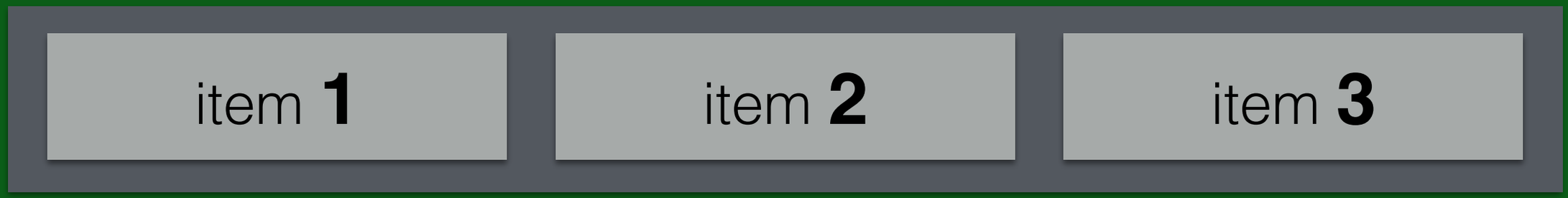


flex-wrap

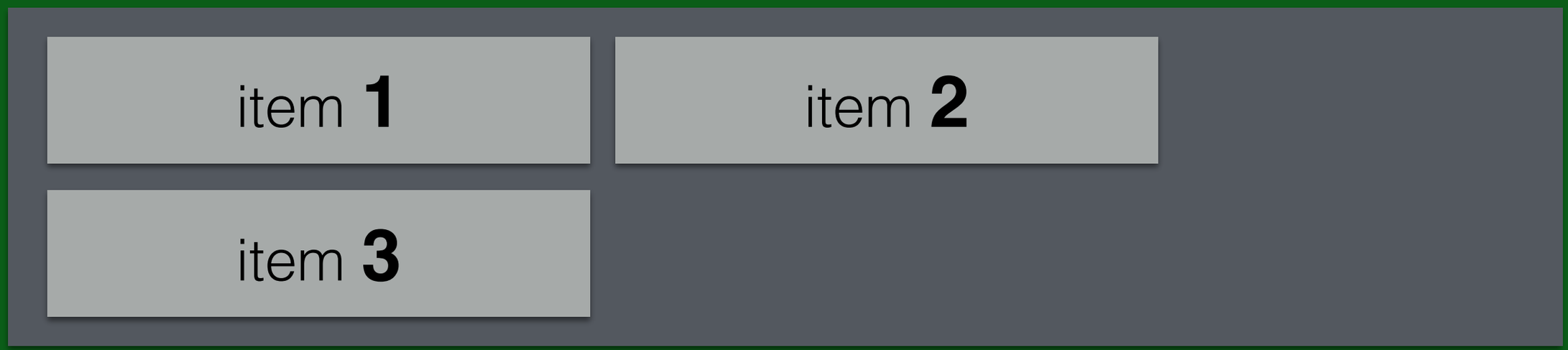
Define if the flex items will break onto multiple lines if their width are larger than width of container.

nowrap

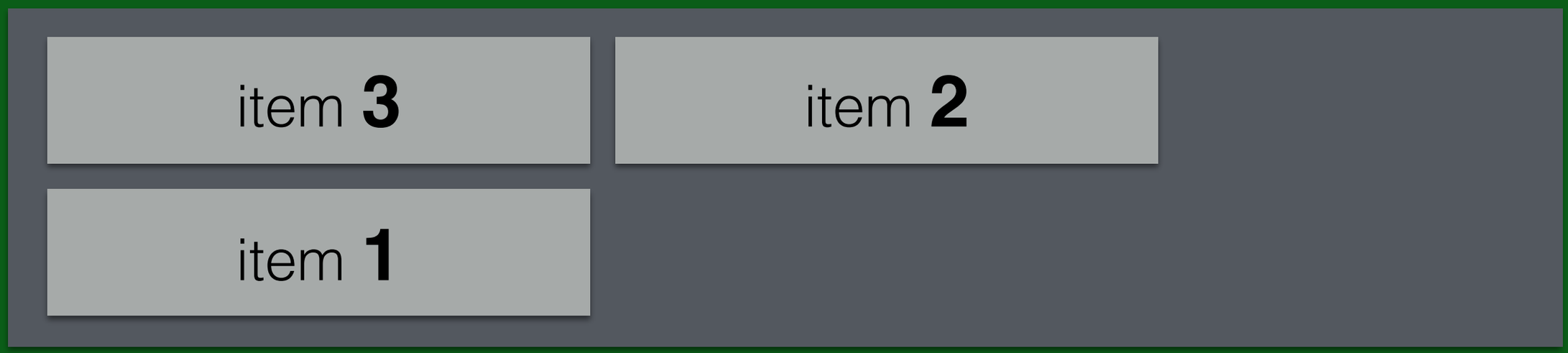
default



wrap



wrap-reverse



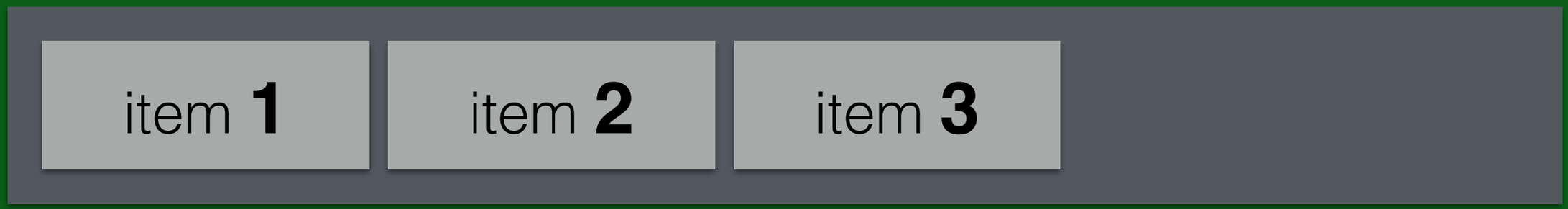
justify-content

Determine align of flex items in main-axis (**horizontal line**).

justify-content

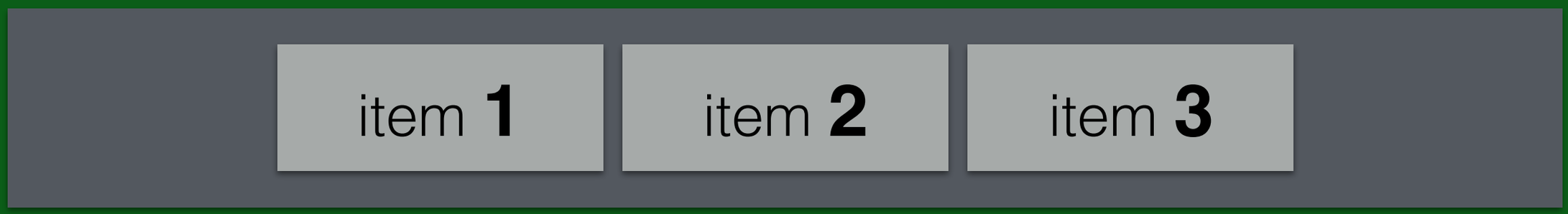
flex-start

default

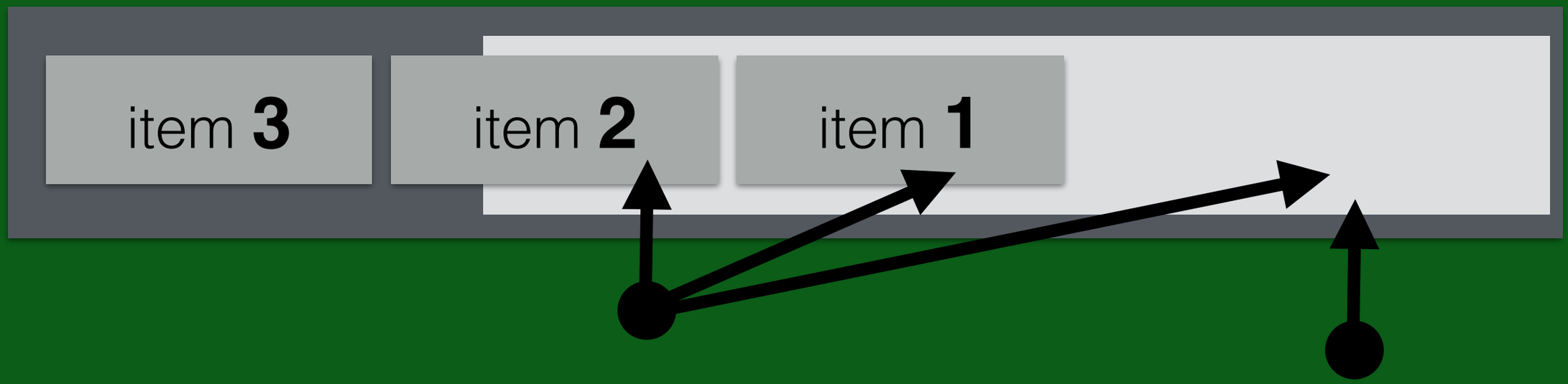


flex-end

center



with float…



justify-content

space-around

item **1** item **2** item **3**

space-between

item **1** item **2** item **3**

align-items

Determine align of flex items in cross-axis (**vertical line**).

align-items

stretch

default

item **1**

item **2**

item **3**

center

item **1**

item **2**

item **3**

align-items

flex-start

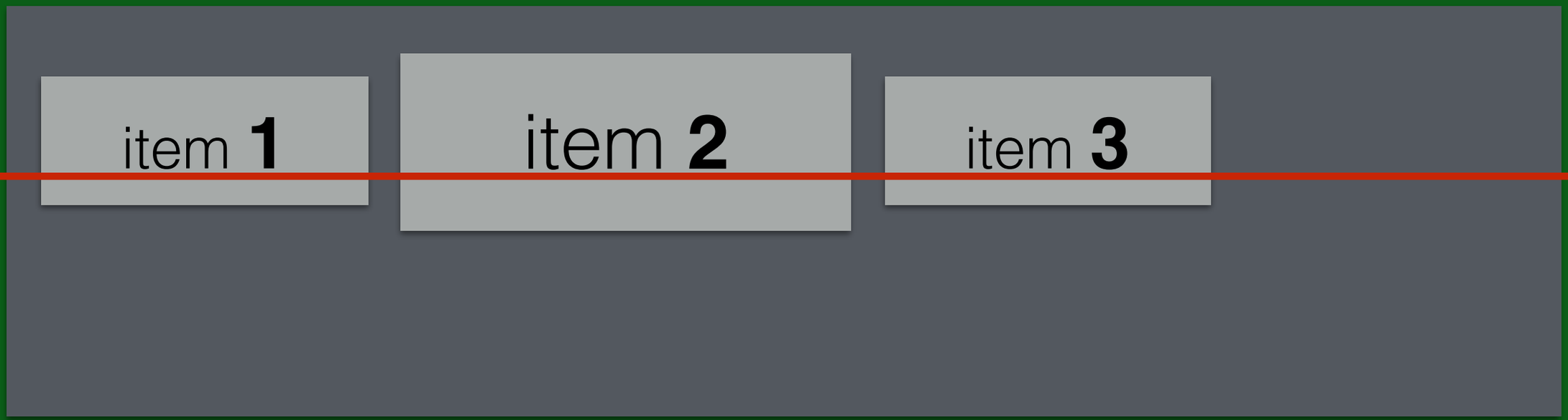
item **1** item **2** item **3**

flex-end

item **1** item **2** item **3**

align-items

baseline

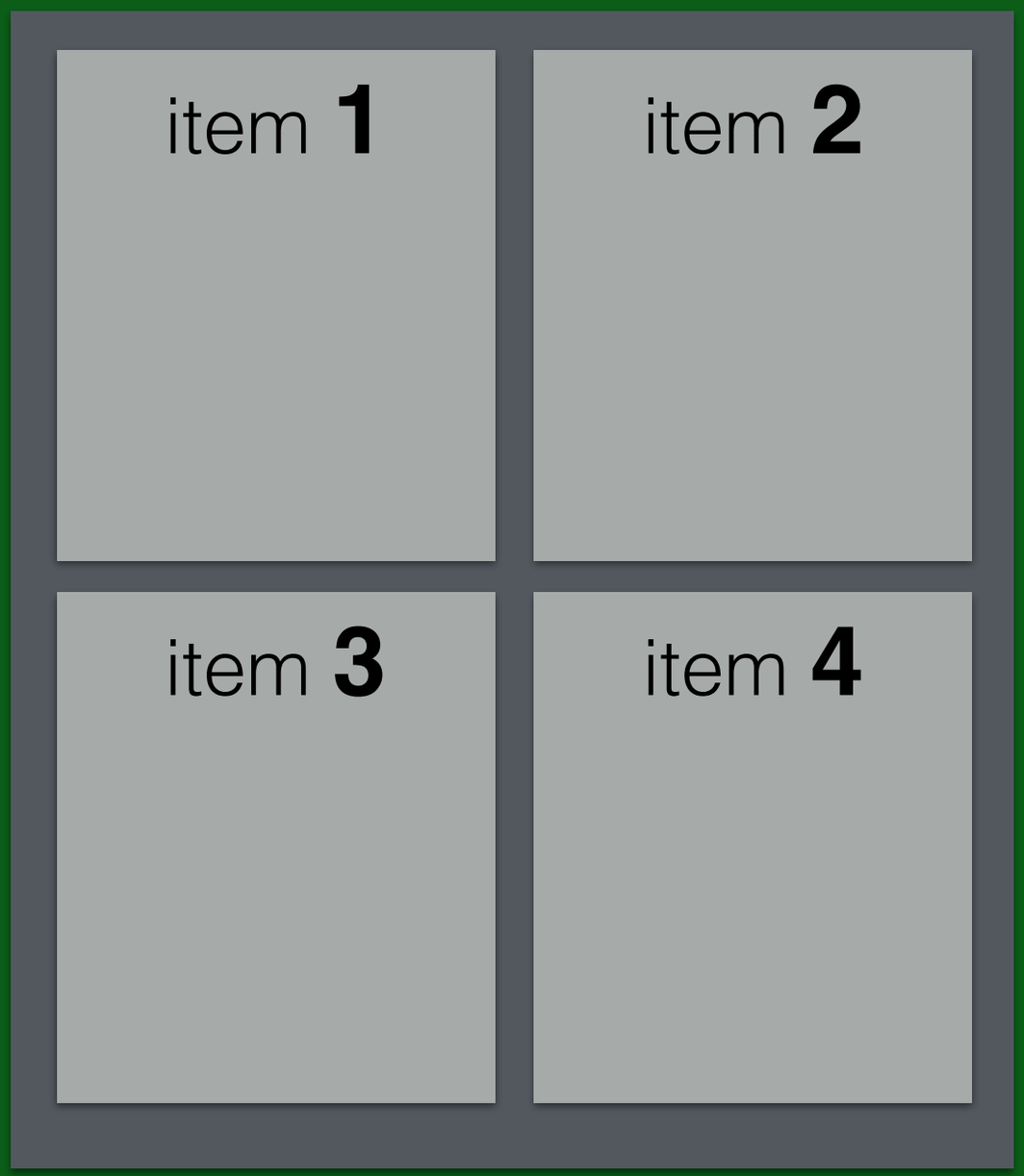


align-content

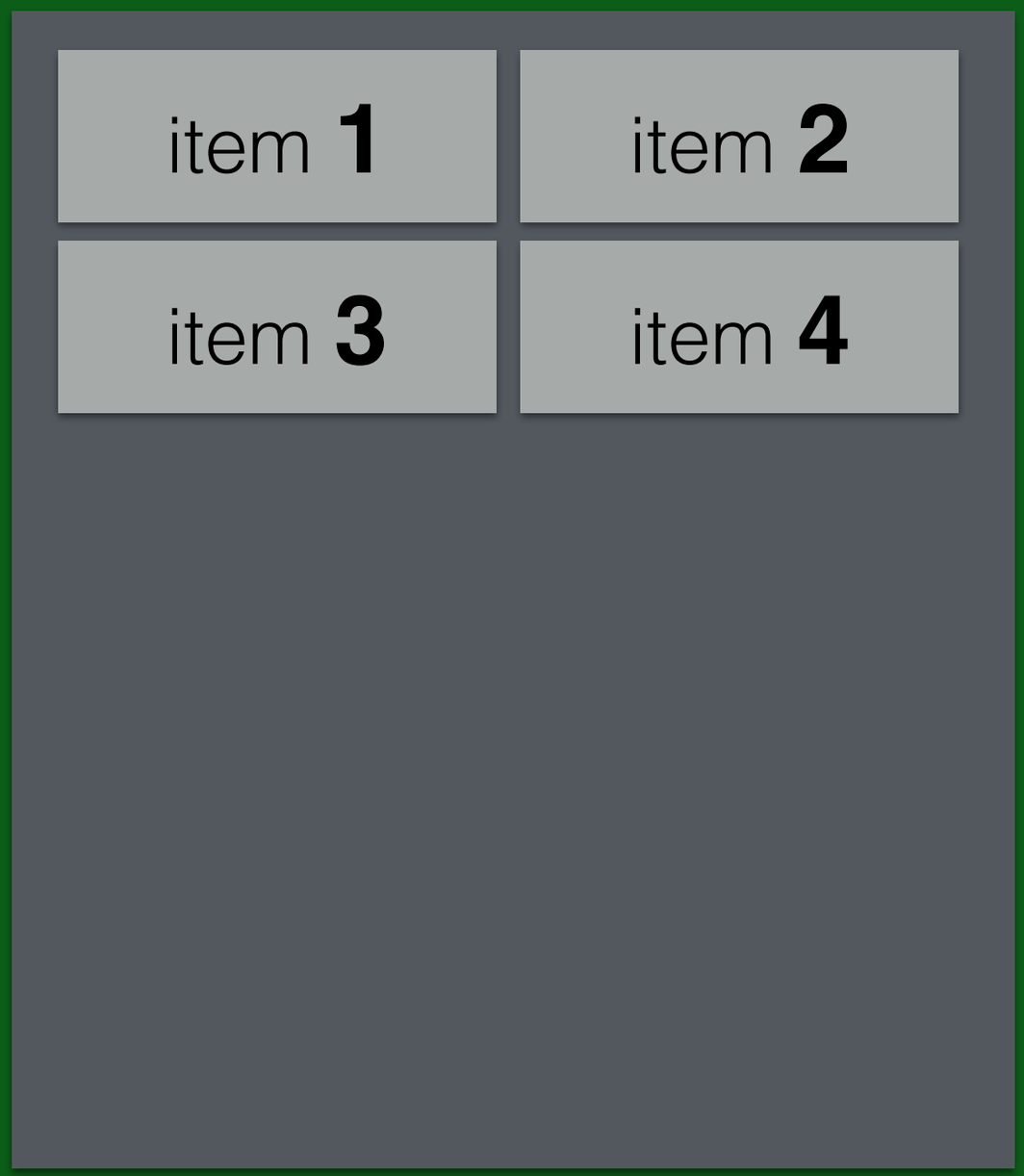
Align flex items with extra space on the **cross-axis,** within the flex container **when have multiple lines**.

align-content

stretch



flex-start



align-content

flex-end

item **1**

item **2**

item **3**

item **4**

center

item **1**

item **2**

item **3**

item **4**

align-content

space-around

item **1**

item **2**

item **3**

item **4**

space-between

item **1**

item **2**

item **3**

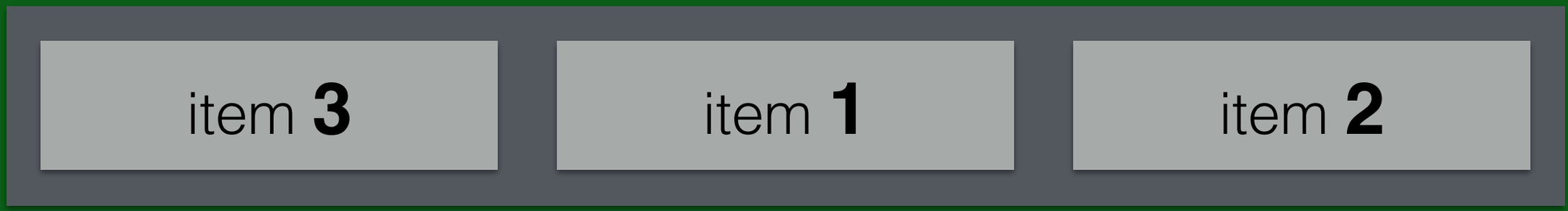
item **4**

Flex Items

order

Man, this is magic. This is awesome. This is “chuchu beleza”.

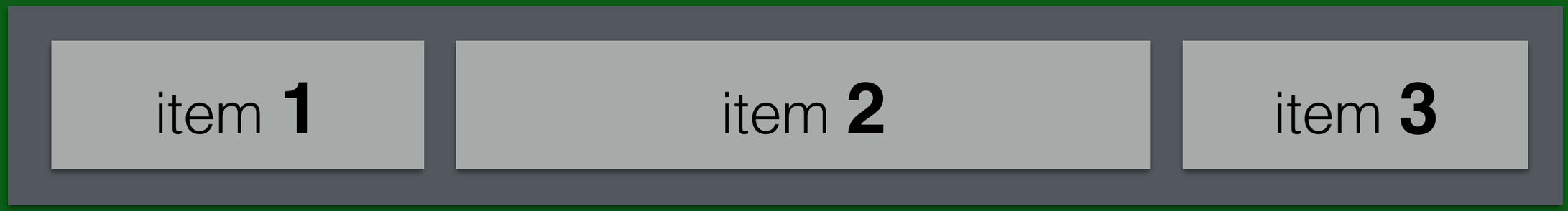
```
.item1 { order: 2; }    .item2 { order: 3; }    .item3 { order: 1; }
```



flex-grow

Define how much the item will take of available space. The value serves as a proportion. If all elements have 1 of value, all elements will have same width. If one element have 2 of value, that element will have the double of size.

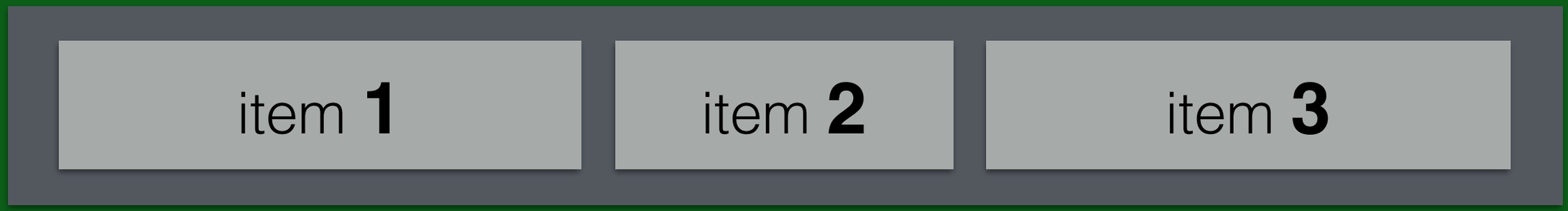
```css
.item2 { flex-grow: 2; }
```



flex-shrink

Define how much the item will shrink.

```css
.item2 { flex-shrink: 2; }
```

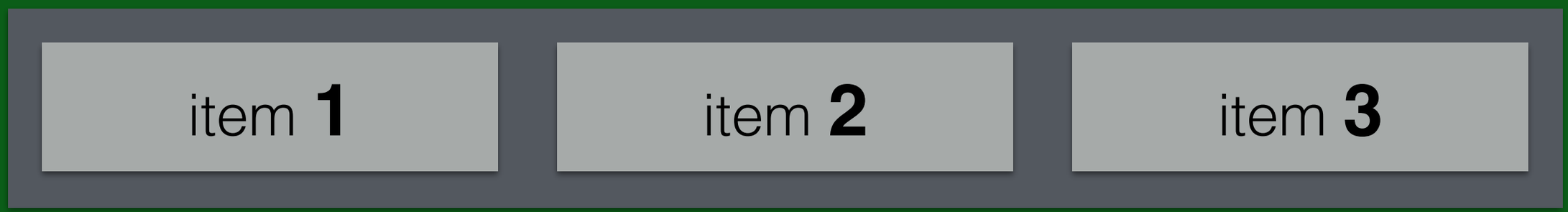


flex-basis

Define the width of elements. This is works like max-width.

```
.item { flex-basis: 100px; }
```

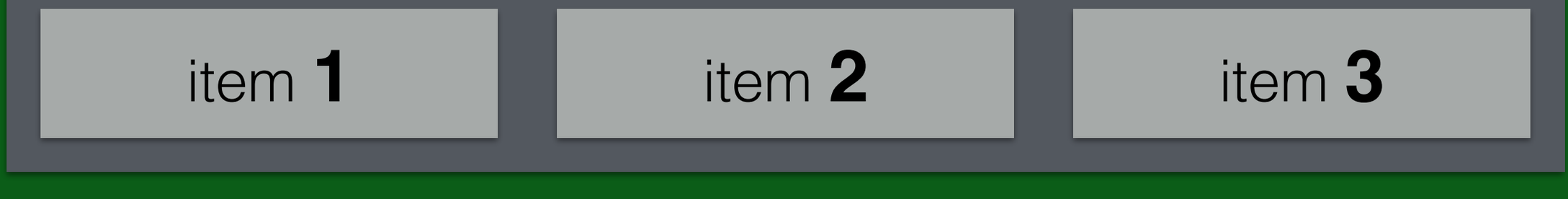

```
.item { flex-basis: 100%; }
```



flex

Shorthand that make all the magic.

```css
.item { flex: 1; }
```




```css
.item {
  flex-grow: 1;
  flex-shrink: 1;
  flex-basis: auto;
}
```

DEMO

IE	Firefox	Chrome	Safari
		31	
		36	
		37	
8		39	
9	31	40	
10	36	41	7
11	37	42	8
Edge	38	43	
	39	44	
	40	45	

http://flexiejs.com

philipwalton.github.io/solved-by-flexbox

Goodbye!

Let me know what you think!

@diegoeis

@tableless

http://**tableless.com.br**
http://**medium.com/@diegoeis**
http://**slideshare.net/diegoeis**