

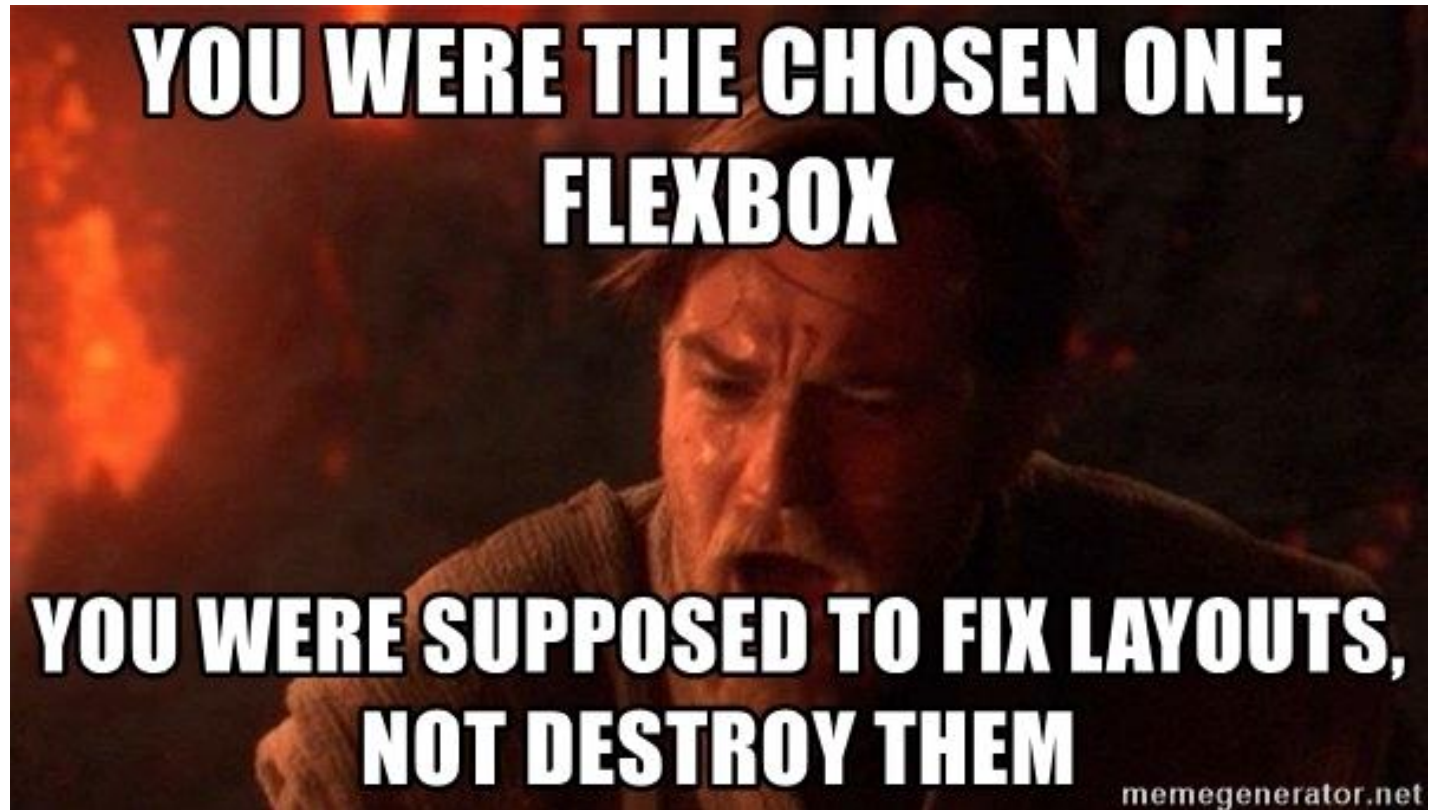
Humanity's victories:



land probe perfectly on a comet 310 million miles away, using science



get stuff on a web page to align properly using CSS

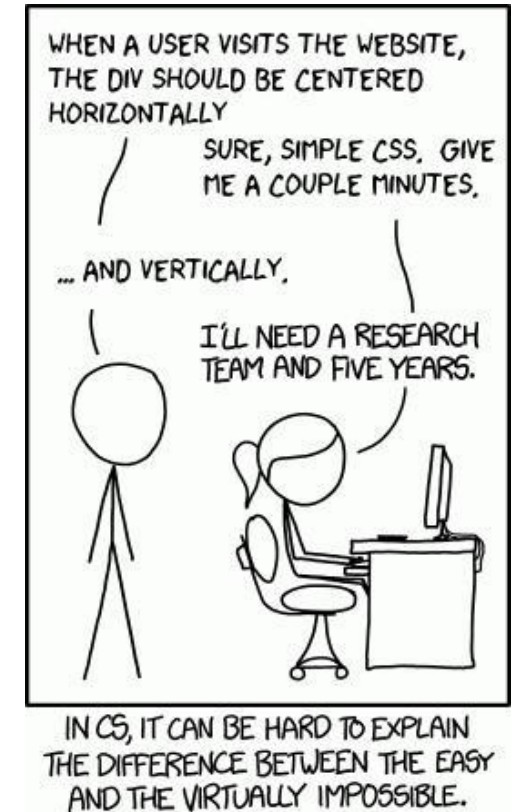


# Module 3 - 4

CSS Flexbox

# Objectives

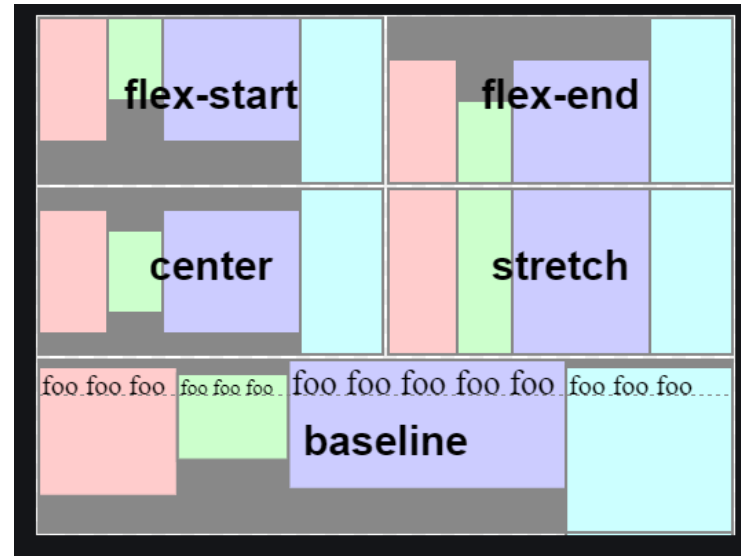
- What is Flexbox
- Define a Flexbox container using a row or a column
- Apply normal flow to Flexbox items
- Apply content alignment to flex items
- Arrange items
- Size items
- Understand how to add Flexbox layouts to existing Responsive CSS Grid layouts
- Understand when to use Flexbox or Grid or combine them both



# Flexbox: Introduction

Flexboxes are specialized containers that have within them several HTML elements arranged in a specific fashion.

The flexbox will also automatically adjust to external stimuli (i.e. resizing the window).

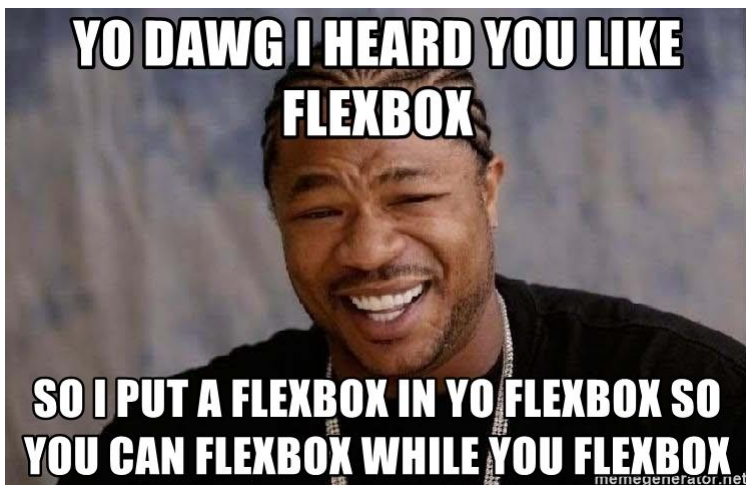


# Flexbox: Defining a container

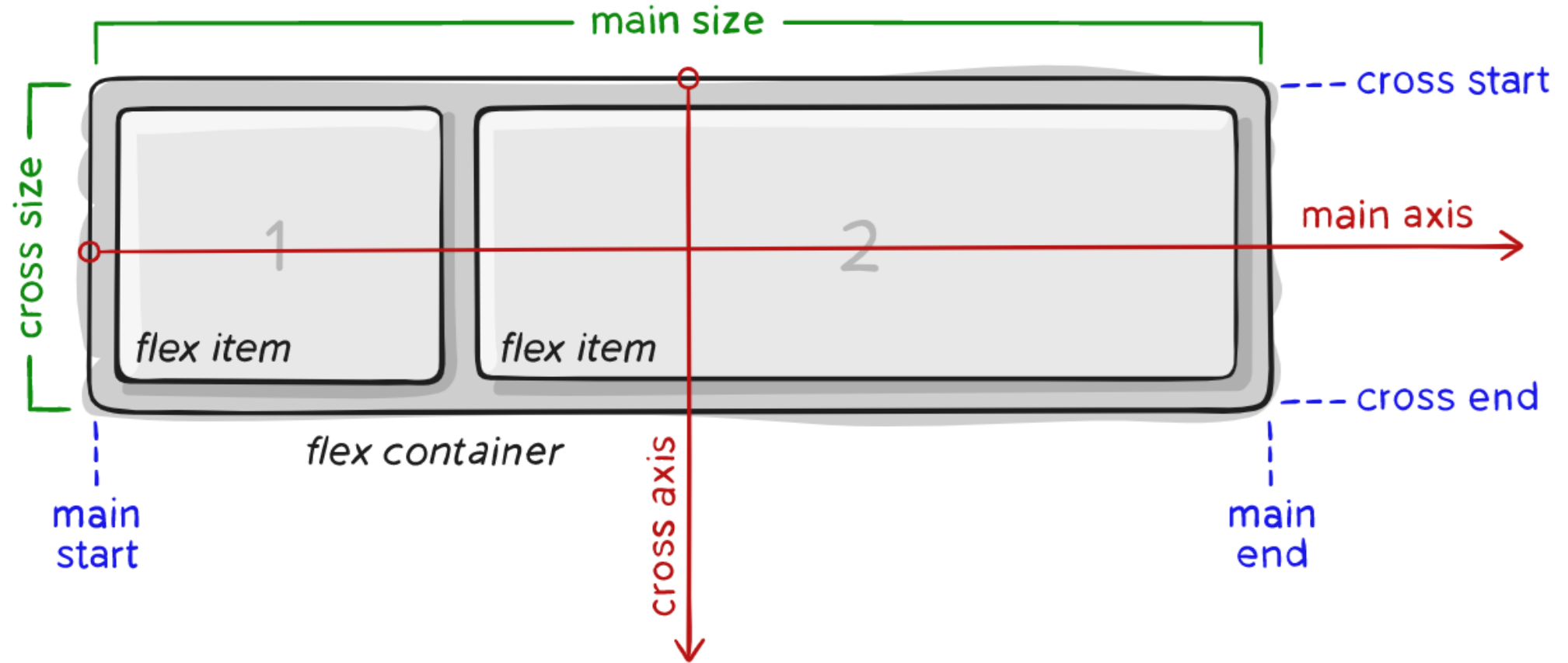
To define a flexbox we must specify a display attribute with a value of grid:

```
nav {  
    display: flex;  
}
```

Here, all HTML elements of type nav will be defined as flex box containers.



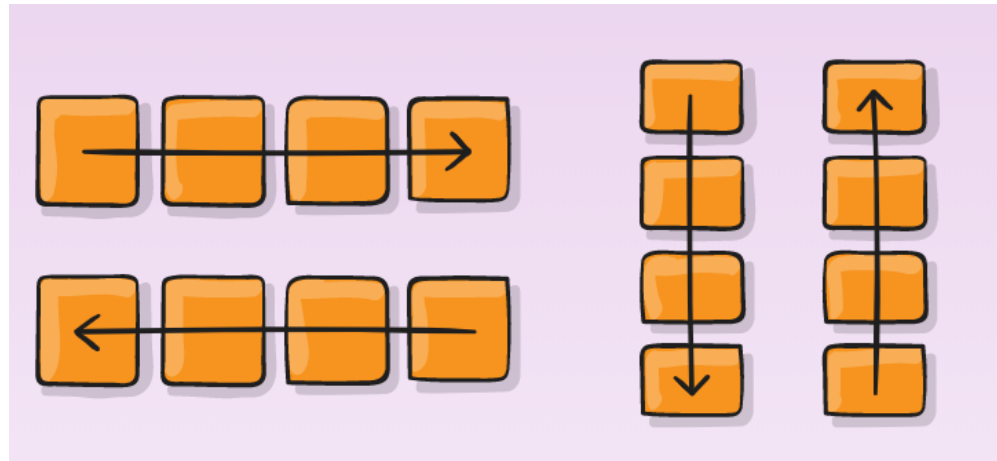
# Flexbox: Anatomy



# Flexbox: flex-direction

The flex-direction attribute determines whether or not the layout of items inside the box will be in either column or rows. **The default direction is row.**

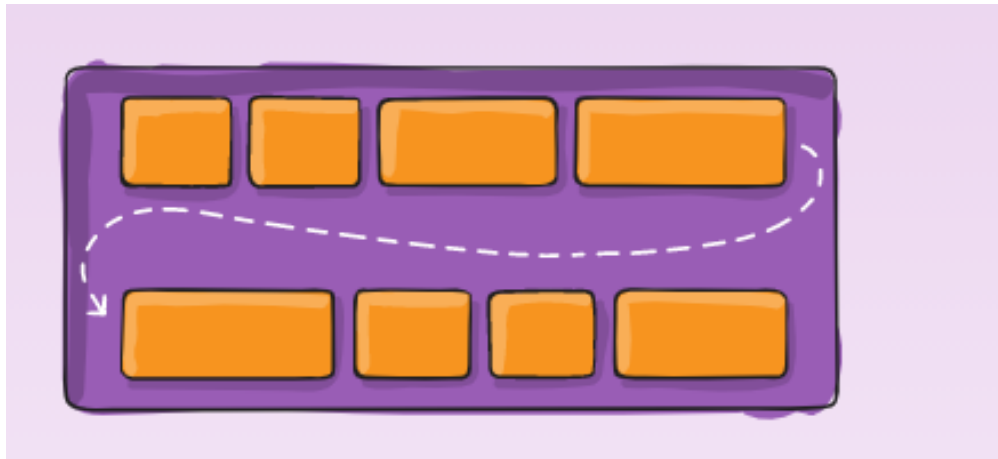
```
nav {  
    display: flex;  
    flex-direction: column;  
}
```



# Flexbox: flex-wrap

The flex-wrap attribute determines whether items will fit all on one line, wrap, or reverse wrap (from bottom to top). **The default is nowrap.**

```
nav {  
    display: flex;  
    flex-wrap: wrap;  
}
```





# Flexbox: flex-flow

The flex-flow attribute is shorthand for flex-direction and flex-wrap. **The default is row nowrap.**

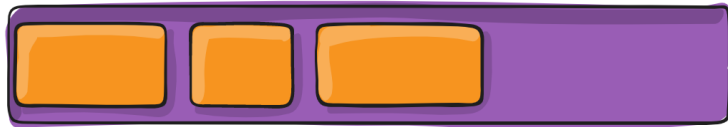
```
nav {  
    display: flex;  
    flex-flow: column wrap;  
}
```

# Flexbox: justify-content

The justify-content property defines how items are aligned across the main axis.

```
nav {  
    display: flex;  
    justify-content: flex-start;  
}
```

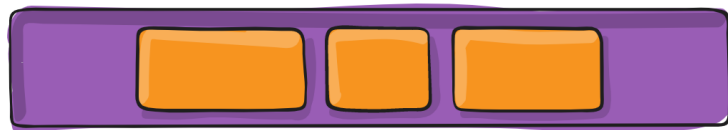
flex-start



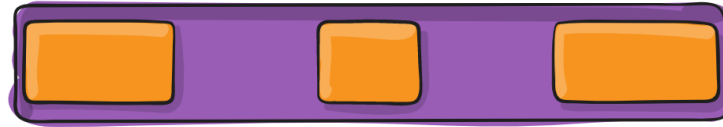
flex-end



center



space-between



space-around



space-evenly

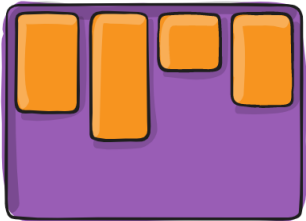


# Flexbox: align-items

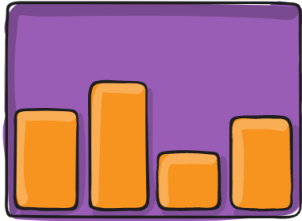
The justify-content property defines how items are aligned across the main axis.

```
nav {  
  display: flex;  
  align-items: flex-start;  
}
```

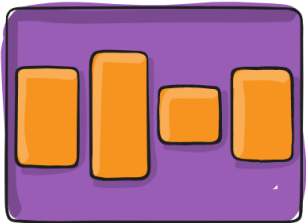
flex-start



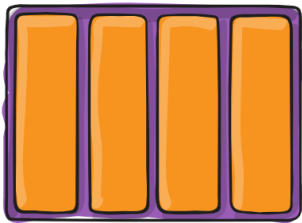
flex-end



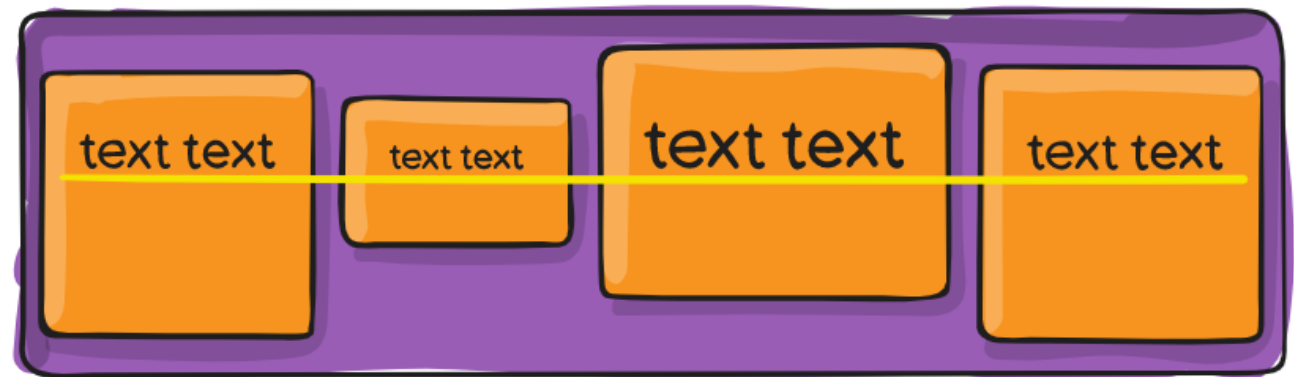
center



stretch



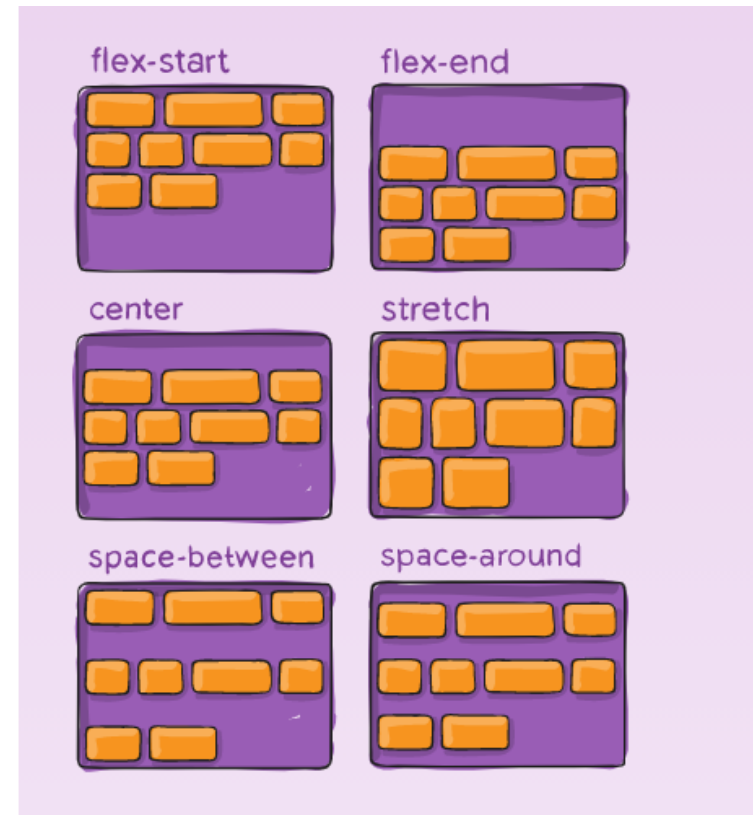
baseline



# Flexbox: align-content

The align-content aligns a flex container's lines within when there is extra space in the cross-axis. This only takes effect on multi-lines flexible containers.

```
nav {  
  display: flex;  
  align-items: flex-start;  
}
```

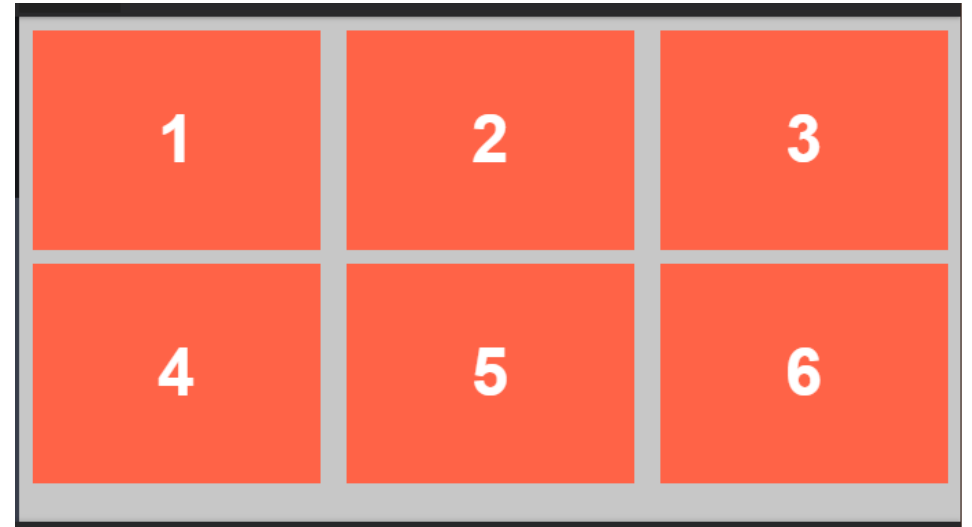


# Flexbox: Defining a child

A child is any element within the flex container.

```
ul {  
  display: flex;  
  flex-flow: row wrap;  
}  
li {  
  background: tomato;  
  padding: 5px;  
}
```

```
<ul>  
  <li>1</li>  
  <li>2</li>  
  <li>3</li>  
  <li>4</li>  
  <li>5</li>  
  <li>6</li>  
</ul>
```



# Flexbox child property: order

By default, flex items are laid out in source order. The `order` property allows you to control the order in which they appear in the flex container. The first position is position 0.



```
.box {  
  display: flex;  
  flex-direction: row;  
}  
.box :nth-child(1) { order: 2; }  
.box :nth-child(2) { order: 3; }  
.box :nth-child(3) { order: 1; }  
.box :nth-child(4) { order: 3; }  
.box :nth-child(5) { order: 1; }
```

```
<div class="box">  
  <div><a href="#">1</a></div>  
  <div><a href="#">2</a></div>  
  <div><a href="#">3</a></div>  
  <div><a href="#">4</a></div>  
  <div><a href="#">5</a></div>  
</div>
```

Reset

# Flexbox child property: flex-grow

The flex-grow property defines the ability for a flex-item to grow if necessary.

```
.item {  
  
}
```

`flex-grow: 4;`

This child will be given 4 times as much space as others

## flex-grow



```
article:nth-child(1),  
article:nth-child(4) {  
  flex-grow: 2;  
}  
article:nth-child(2),  
article:nth-child(3) {  
  flex-grow: 1;  
}
```


AAAAAAA BBBB CCCC DDDDDDD

# Flexbox child property: flex-shrink

The flex-shrink property defines the ability for a flex-item to shrink if necessary. You remove space rather than add.

```
.item {  
  
}
```

```
flex-shrink: 2;
```



This child will be given 2 times less space as others



# Flexbox child property: flex-basis

The flex-basis property defines the default size of an element before the remaining space is distributed.

```
.item {  
    flex-basis: 20rem;  
}
```

# Flexbox child property: flex

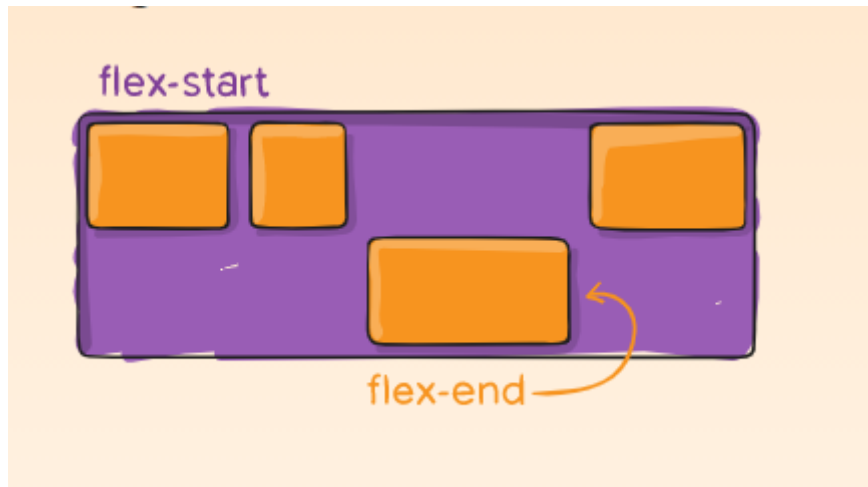
The flex property is shorthand for flex-grow, flex-shrink and flex-basis combined. The default is 0 1 auto. This is recommended over the individual properties.

```
.item {  
    flex: 2 2 20rem;  
}
```

# Flexbox child property: align-self

The align-self property allows the default alignment to be overridden for individual flex items.

```
.item {  
    align-self: flex-end;  
}
```



Let's do some coding!