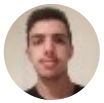
# CSS Flexbox Explained With Examples

Learn about CSS flexbox with practical examples.



Mehdi Aoussiad Following

Jan 2 · 5 min read ★


## Introduction

Flexbox is one of the greatest features that has been added to CSS. It was designed as a one-dimensional layout model, and as a method that could offer space distribution between items in an interface and powerful

alignment capabilities. Flexbox makes it easier to design flexible responsive layouts structure without using float or positioning.

In this article, we will learn about CSS flexbox with some practical examples. Let's get right into it.

## Define a container

To start using flexbox, you will need to define a container `div` or a parent `div` where you will wrap all the child elements like this:

```
<div class="container">
  <div>1</div>
  <div>2</div>
  <div>3</div>
</div>
```

The parent `div` or the container becomes flexible by setting the `display` property to *flex*:

```
.container {
  display: flex;
```

```
  background-color: red;
}

.container div{
  background-color: #f1f1f1;
  margin: 10px;
  padding: 20px;
  font-size: 30px;
}
```

Here is the output:


A red container that contains three flexible divs.

As you can see, by setting the property `display` to *flex,* the child elements of the container automatically become flexible items. Now you can use the container properties such as `justify-content align-items` for example in order to center your child elements inside the container div. We will cover that in the examples below.

## The flex-direction property

The `flex-direction` property defines in which direction the container wants to stack the flex items(column or row).

The example below sets the flex-direction to `column` (from top to bottom). As a result, the child elements inside your container div will be in a vertical line.

Have a look at the examples below:

```
.container {
  display: flex;
  flex-direction: column;
  background-color: red;
}

.container > div {
  background-color: #f1f1f1;
  width: 100px;
  margin: 10px;
  text-align: center;
  line-height: 75px;
  font-size: 30px;
}
```

Here is the output:


Column direction.

Here is the same example but we set the property `flex-direction` to `row` which will make the child elements in a horizontal line inside our container:

```
.container {
  display: flex;
  flex-direction: row;
  background-color: red;
}
```

```
.container > div {
  background-color: #f1f1f1;
  width: 100px;
  margin: 10px;
  text-align: center;
  line-height: 75px;
  font-size: 30px;
}
```

Output:


Row direction.

You can also reverse the order of the child elements inside the container by setting the property `flex-direction` to `column-reverse` or `row-reverse` .

## The flex-wrap property

The property `flex-wrap` specifies whether the flex items should wrap or not.

The example below has 12 flex items and sets the property `flex-wrap` to `wrap` . To better demonstrate the `flex-wrap` property. I would recommend that you put the code below in your text editor or in Codepen and resize the browser window to see the power of `flex-wrap` .

```
HTml:
<div class="container">
  <div>1</div>
  <div>2</div>
  <div>3</div>
  <div>4</div>
  <div>5</div>
  <div>6</div>
  <div>7</div>
  <div>8</div>
  <div>9</div>
  <div>10</div>
  <div>11</div>
  <div>12</div>
</div>


CSS:
.container {
  display: flex;
  flex-wrap: wrap;
  background-color: red;
}


.container > div {
  background-color: #f1f1f1;
  width: 100px;
  margin: 10px;
```

```
  text-align: center;
  line-height: 75px;
  font-size: 30px;
}
```

If you want the child elements or the flex-items to not wrap, you can set the property `flex-wrap` to `nowrap` like this:

```
.container {
  display: flex;
  flex-wrap: nowrap;
  background-color: red;
}
```

## The justify-content property

The `justify-content` property is used to align the flex items. You can give this property some values such as: `center` , `flex-start` , `flex-end` , `space-between` and etc.

The *center* value aligns the flex items at the center of the container:

```
.container {
  display: flex;
  justify-content: center;
}
```


The *flex-start* value aligns the flex items at the beginning of the container:

```
.container {
  display: flex;
  justify-content: flex-start;
}
```

The *flex-end* value aligns the flex items at the end of the container:

```
.container {
  display: flex;
  justify-content: flex-end;
}
```


The *space-between* value displays the flex items with space between the lines:

```
.container {
  display: flex;
  justify-content: space-between;
}
```


## The align-items property

The `align-items` property is used to align the flex items. It's the same as `justify-content` but we work vertically instead of horizontally. That's why I will only give one example instead of repeating the same examples.

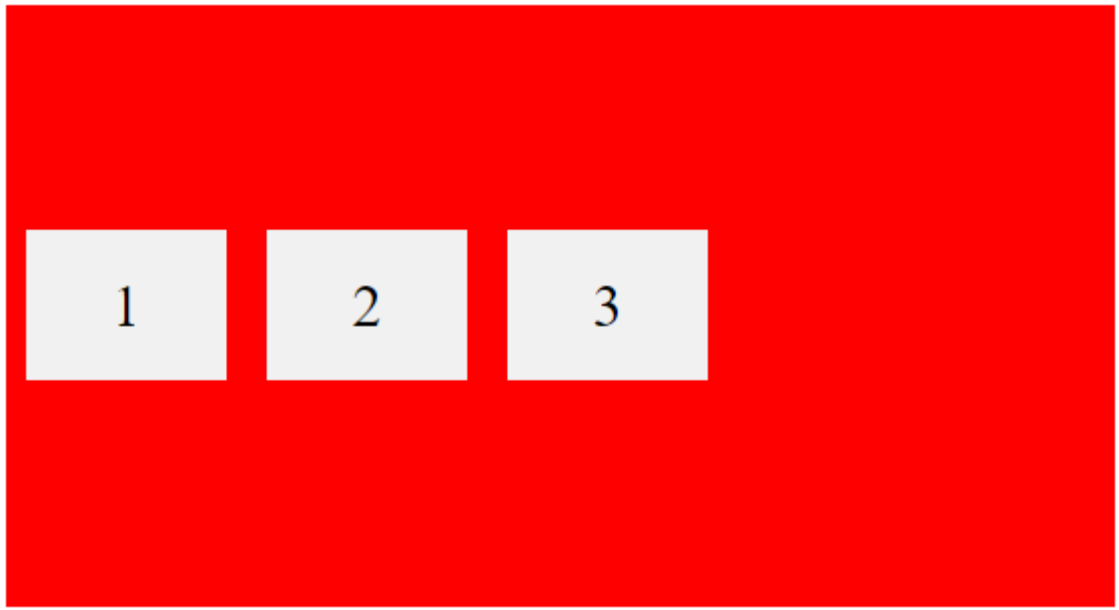
So here is an example that centers the child elements vertically inside the container:

```
.container {
  display: flex;
  height: 300px;
  align-items: center;
  background-color: red;
}

.container > div {
  background-color: #f1f1f1;
  width: 100px;
  margin: 10px;
  text-align: center;
  line-height: 75px;
```

```
  font-size: 30px;
}
```

Output:




Centering the child elements vertically.

The property `align-items` has the same values as `justify-content` . The only difference is that we are working vertically instead of horizontally.

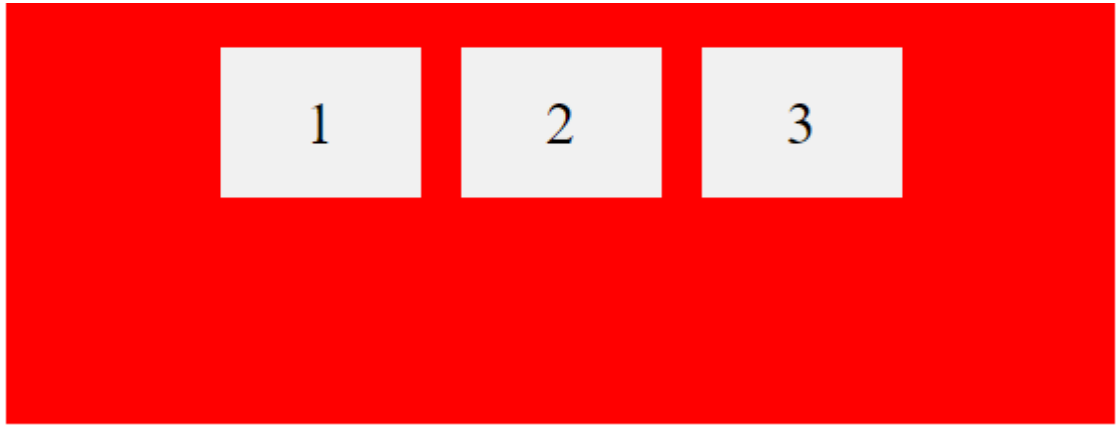
## Centering vertically and horizontally

Now you can use both `justify-content` and `align-items` to easily center your child elements vertically and horizontally.

Here is an example:

```
.container {
  display: flex;
  height: 300px;
  align-items: center;
  justify-content: center;
  background-color: red;
}

.container > div {
  background-color: #f1f1f1;
  width: 100px;
  margin: 10px;
  text-align: center;
  line-height: 75px;
  font-size: 30px;
}
```




Centering vertically and horizontally.

## Child elements

Child elements also have some properties that you can benefit from such as:

- order
- flex-grow
- flex-shrink
- flex-basis
- flex
- align-self

You can learn about them from the W3schools website.

## Conclusion

Flexbox is an awesome CSS feature that allows you to easily design a flexible responsive layout structure. I highly recommend that you practice this feature because practice is the only way to get better at it.

Thank you for reading this article, I hope you found it useful. If so, get more similar content by **subscribing to our YouTube channel!**

## More Reading

The Front-End Web Developer Roadmap for 2021

Step by step guide to becoming a modern front-end web developer.

medium.com