Rebecca Hall
Instructor

CSS Layout: Flexbox

CSS Layout Techniques

Flexible Box Layout Module - Flexbox

CSS Layout Module – offers greater control over arranging components of web page items **along one axis** (Ex: menu bars, product listings, galleries)

Advantages of Flexbox

- Ability to “flex” – stretch or shrink inside their containers
- Ability to make all neighboring items the same height
- Easy horizontal and vertical centering
- Ability to change the display order of items independent of the html source

*Browser Support – older browsers require prefixes/additional code for flexbox to work properly

CSS Layout - Flexbox

Flexbox Container

you create a flex container by setting the element's **display** property to **flex**

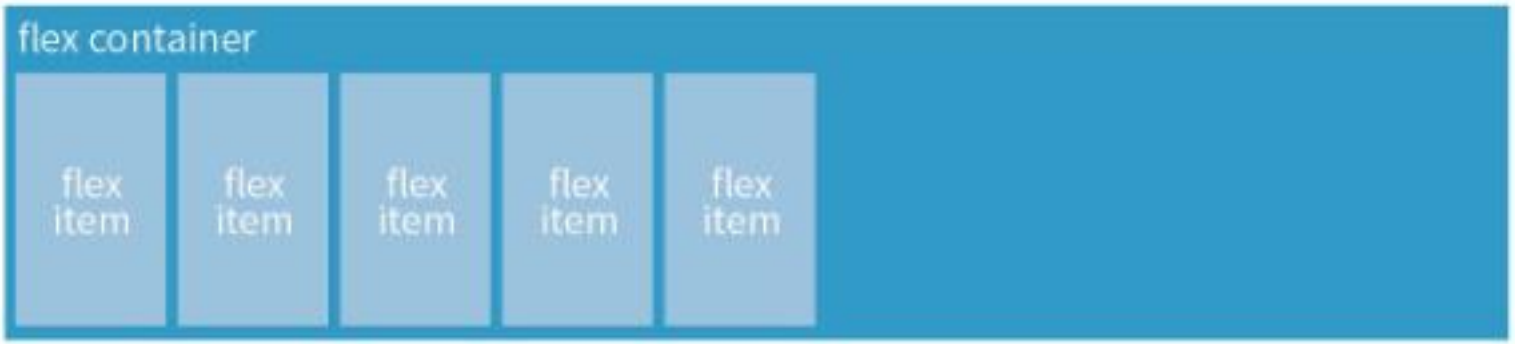


Figure 16-2. Niederst, J. (2018). Learning Web Design. O'Reilly Media, Inc.

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

/* or */

```
.flex-container {
        display: inline-flex;
}
```

Display: flex creates a block-level flex container

Display: inline-flex creates an inline-level flex container

CSS Layout - Flexbox

Flex container - Flex items

applying the flex display mode turns the child elements of the flex container into flex items

Figure 16-2. Niederst, J. (2018). Learning Web Design. O'Reilly Media, Inc.

Note: You can turn any flex item into a flex container by setting its **display** to **flex** (resulting in nested flexbox)

CSS Layout - Flexbox

Flex container - Properties

flex container properties allow you to control how items appear within the container

flex-direction

row | column | row-reverse | column-reverse

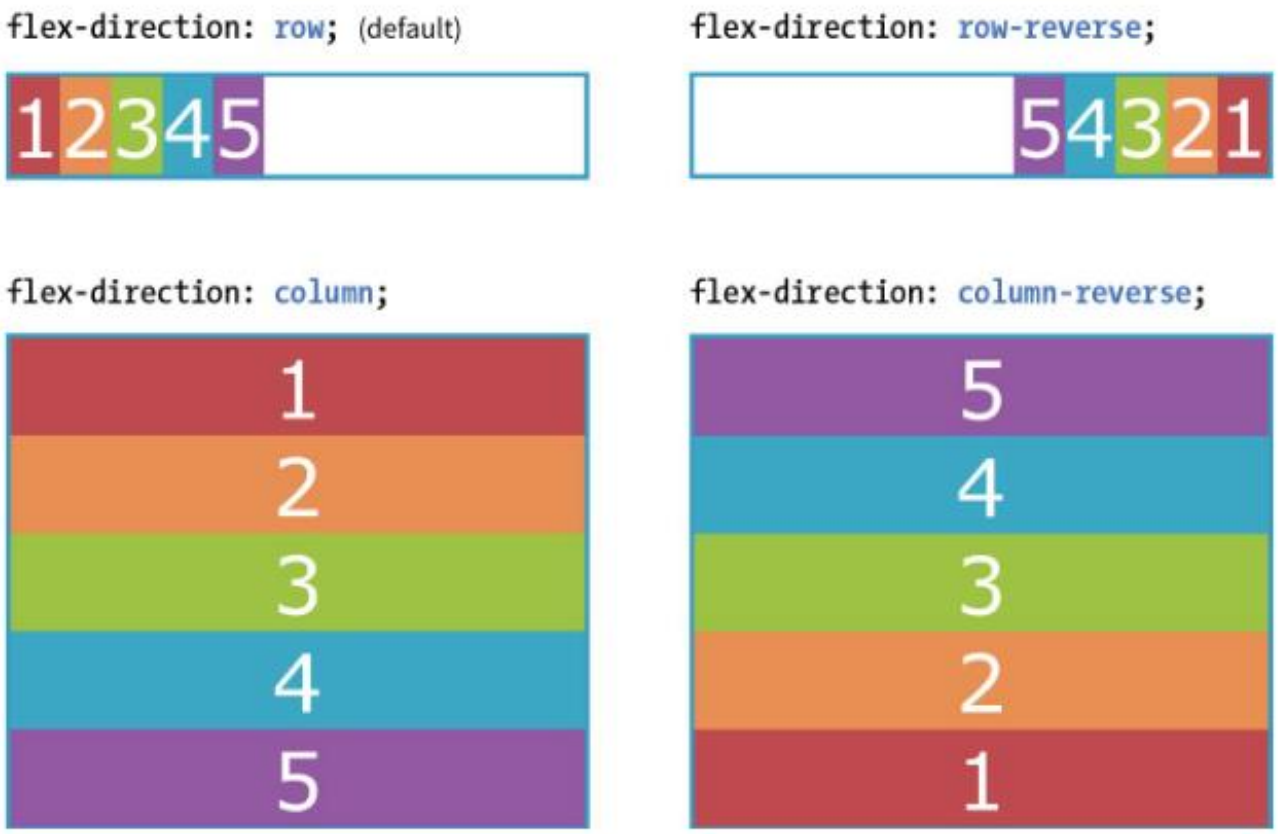
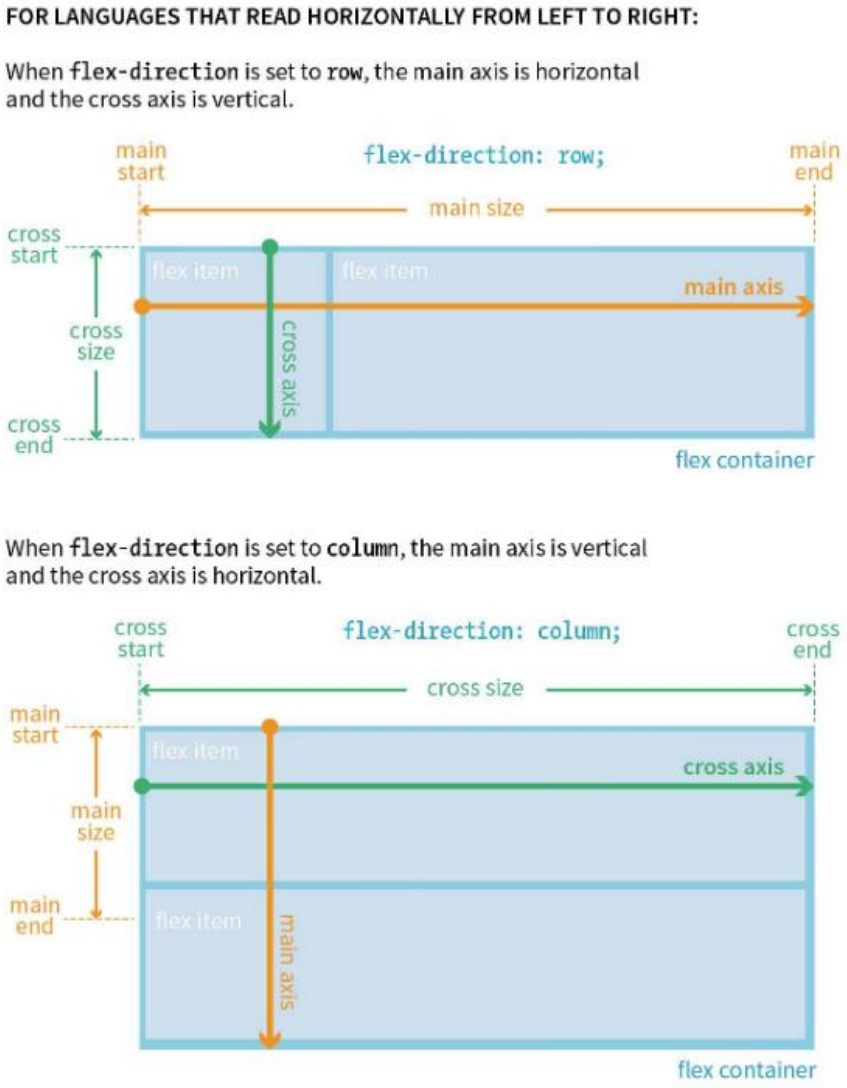


Figure 16-3. Niederst, J. (2018). Learning Web Design. O'Reilly Media, Inc.

CSS Layout - Flexbox



flex items

flex items line up along one axis

main axis -or- cross axis

The main axis is the flow direction you've specified for the flex container. The cross axis is perpendicular to the main axis

Note: axis direction is specific to the direction of the writing system in use.
For example: In horizontally oriented languages – "row" would align items horizontally
Vertically oriented languages – "row" would align items vertically

Figure 16-4. Niederst, J. (2018). Learning Web Design. O'Reilly Media, Inc.

CSS Layout - Flexbox

flex-wrap

nowrap | wrap | wrap-reverse

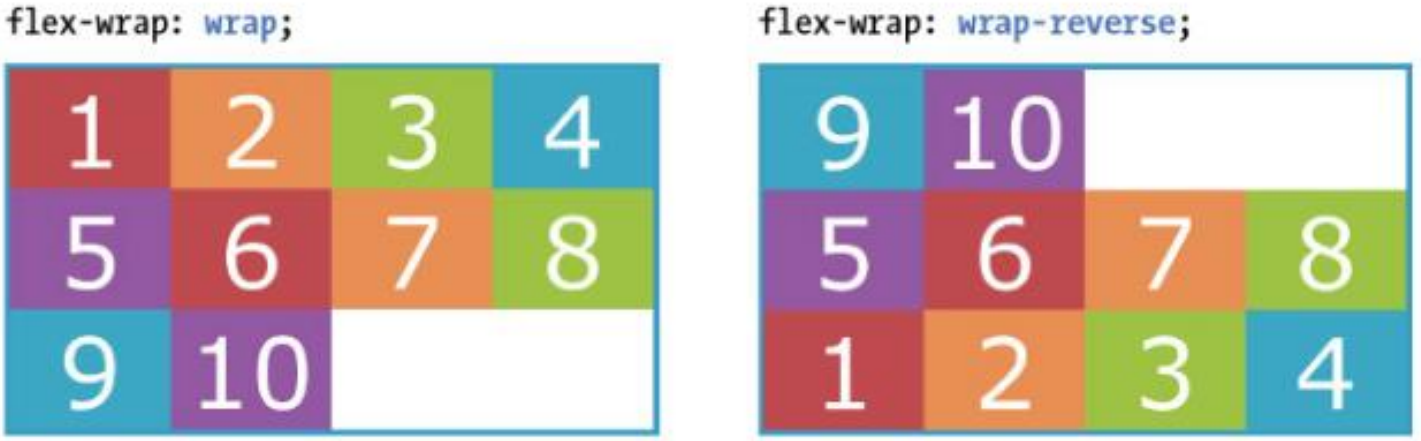


Figure 16-5. Niederst, J. (2018). Learning Web Design. O'Reilly Media, Inc.

THE MARKUP

```
<div id="container">
  <div class="box box1"> 1</div>
  <!– more boxes here →
  <div class="box box10"> 10 </div>
</div>
```

THE STYLES

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

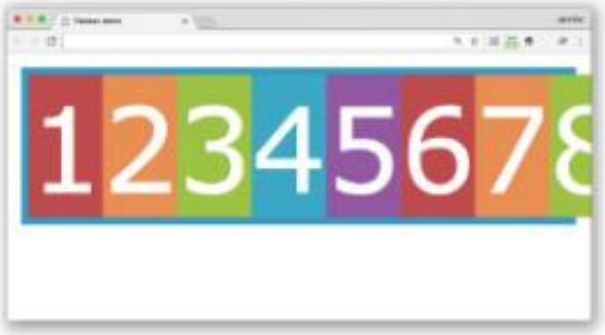
```
.box {
  width: 25%;
}
```

CSS Layout - Flexbox

flex-wrap

nowrap | wrap | wrap-reverse

`flex-wrap: nowrap;` (default)



When wrapping is disabled, flex items squish if there is not enough room, and if they can't squish any further, may get cut off if there is not enough room in the viewport.

Figure 16-5. Niederst, J. (2018). Learning Web Design. O'Reilly Media, Inc.

CSS Layout - Flexbox

flex-wrap

nowrap | wrap | wrap-reverse

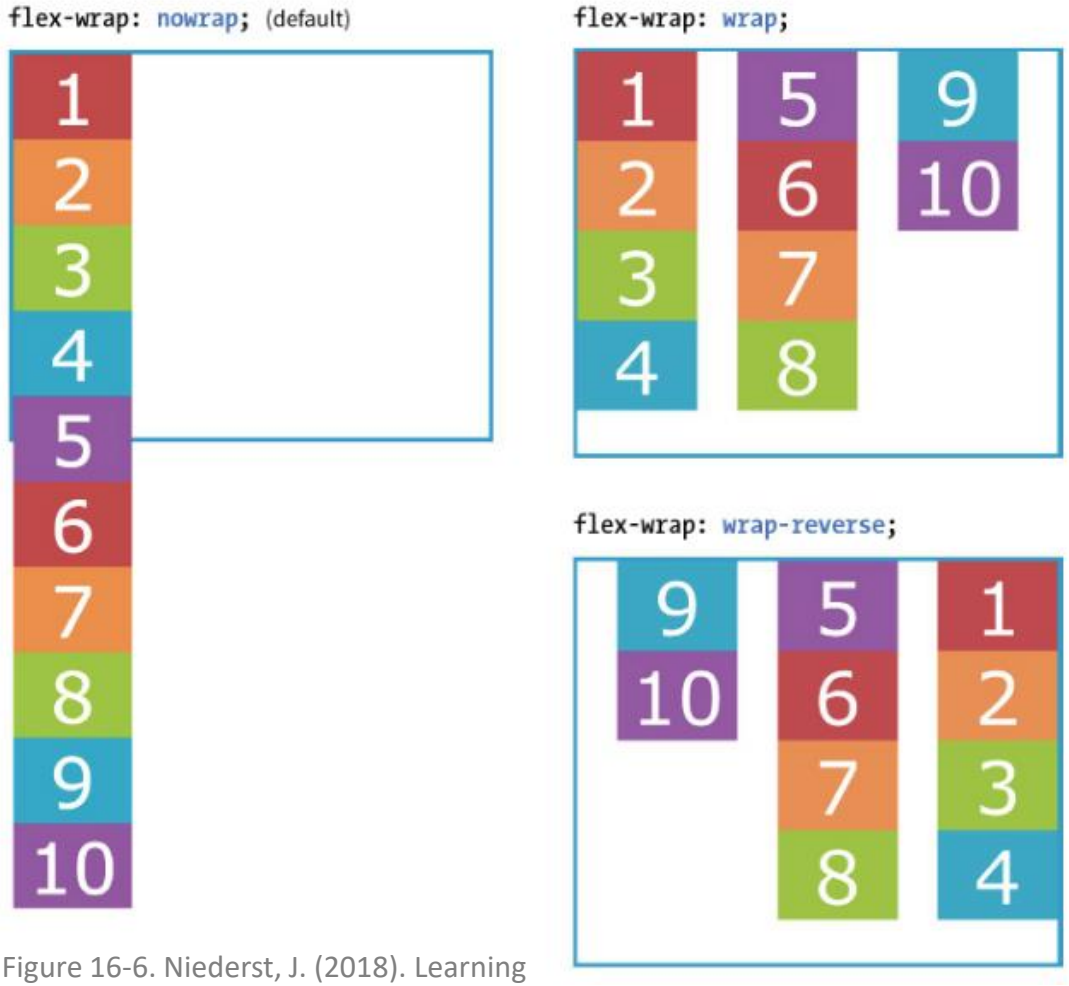


Figure 16-6. Niederst, J. (2018). Learning Web Design. O'Reilly Media, Inc.

THE MARKUP

```
<div id="container">
  <div class="box box1"> 1</div>
  <!– more boxes here →
  <div class="box box10"> 10 </div>
</div>
```

THE STYLES

```
#container {
  display: flex;
  height: 350px;
  flex-direction: column;
  flex-wrap: wrap;
}
  .box {
    width: 25%;
  }
```

CSS Layout - Flexbox

flex-flow (shorthand)

flex-direction flex-wrap

Using flex-direction & flex-wrap

```
#container {
  display: flex;
  height: 350px;
  flex-direction: column;
  flex-wrap: wrap;
}
```

Using **flex-flow**

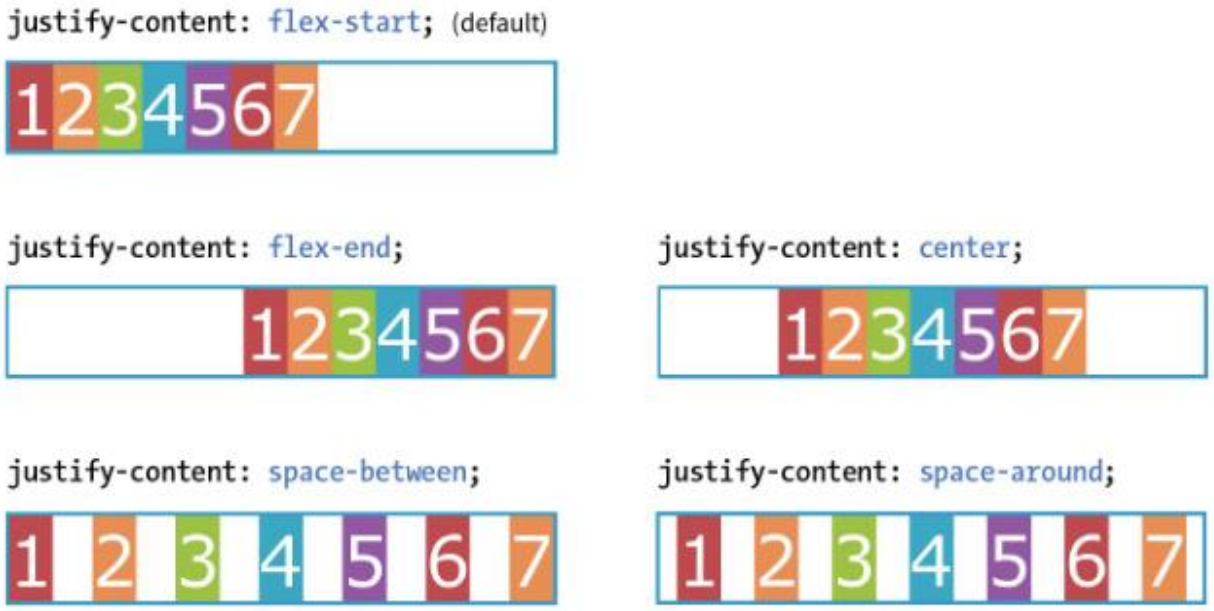
```
#container {
  display: flex;
  height: 350px;
  flex-flow: column wrap;
}
```

CSS Layout - Flexbox

Flexbox Alignment Properties

justify-content

flex-start | flex-end | center | space-between | space-around



Flex items are, by default, as wide as they need to be to contain the element's content.

The **justify-content** property defines how extra space is distributed around or between items

Figure 16-8. Niederst, J. (2018). Learning Web Design. O'Reilly Media, Inc.

CSS Layout - Flexbox

justify-content

flex-start | flex-end | center | space-between | space-around

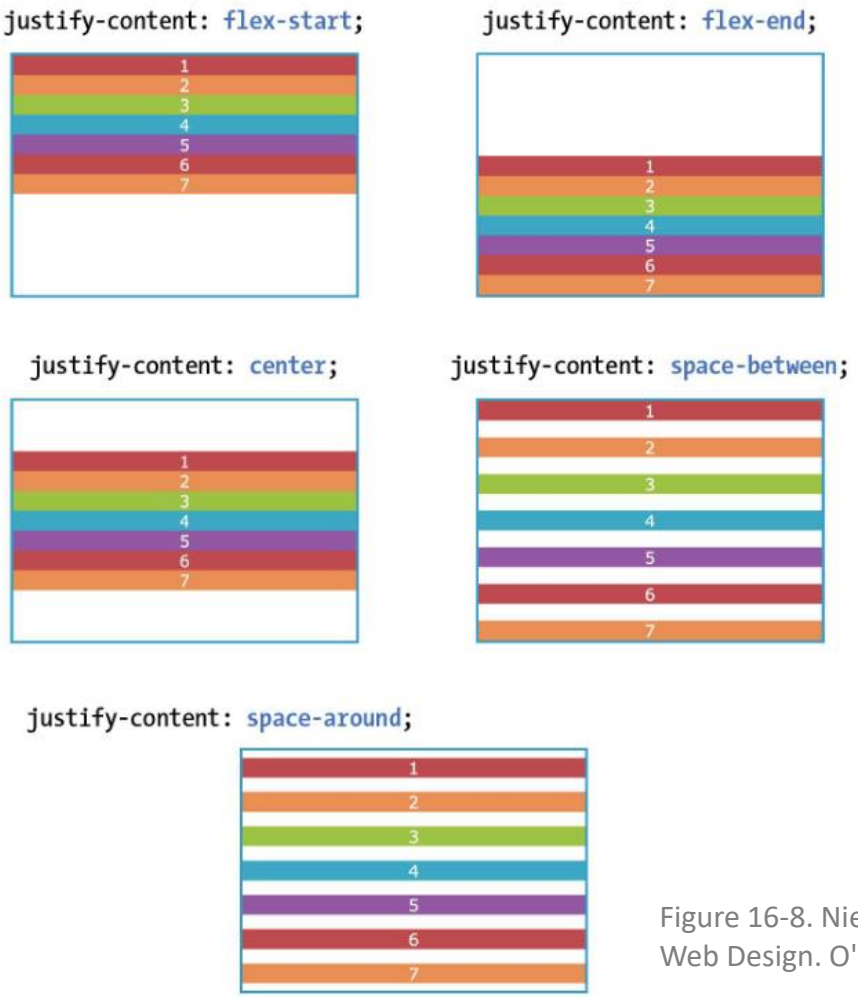


Figure 16-8. Niederst, J. (2018). Learning Web Design. O'Reilly Media, Inc.

Flex items are, by default, as wide as they need to be to contain the element's content.

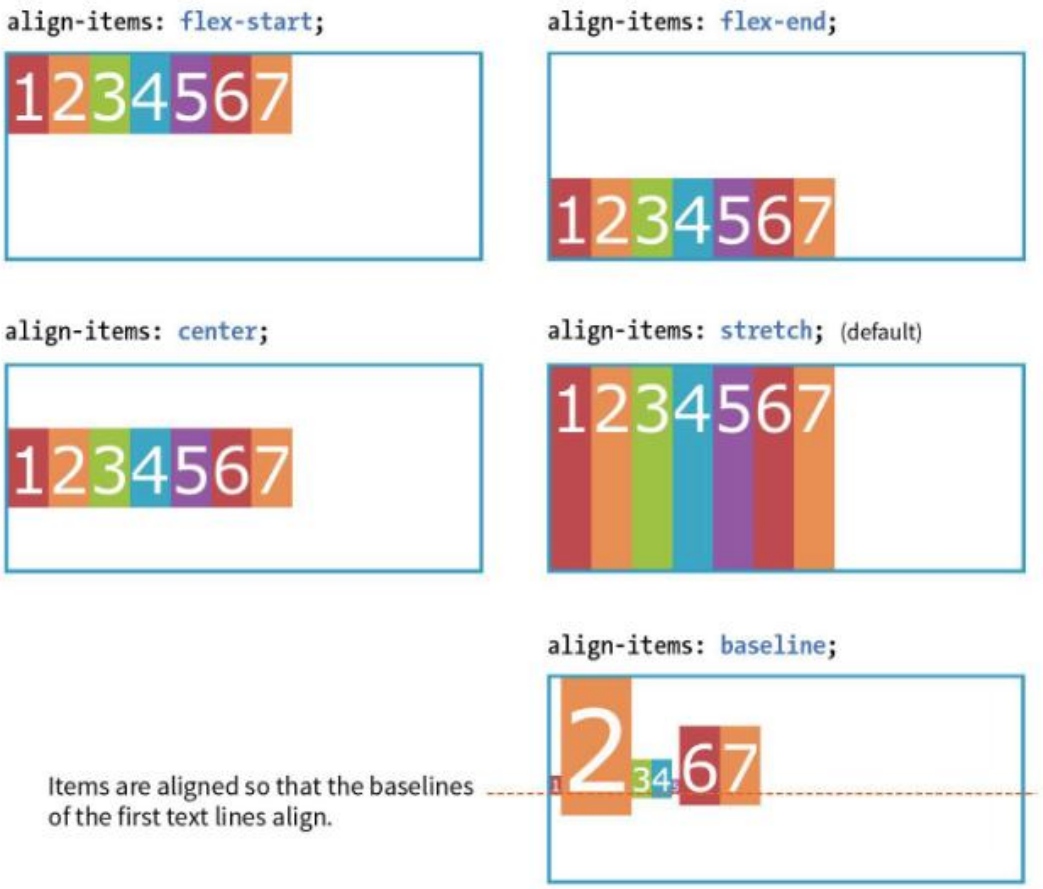
The **justify-content** property defines how extra space is distributed around or between items

CSS Layout - Flexbox

Flexbox Alignment Properties

align-items

flex-start | flex-end | center | baseline | stretch



The **align-items** property allows you to arrange items on the cross axis (up and down when the direction is **row**, left and right if the direction is **column**)

*Note: you must specify a container height

Figure 16-10. Niederst, J. (2018). Learning Web Design. O'Reilly Media, Inc.

CSS Layout - Flexbox

Flexbox Alignment Properties

align-content

flex-start | flex-end | center | space-around | space-between | stretch

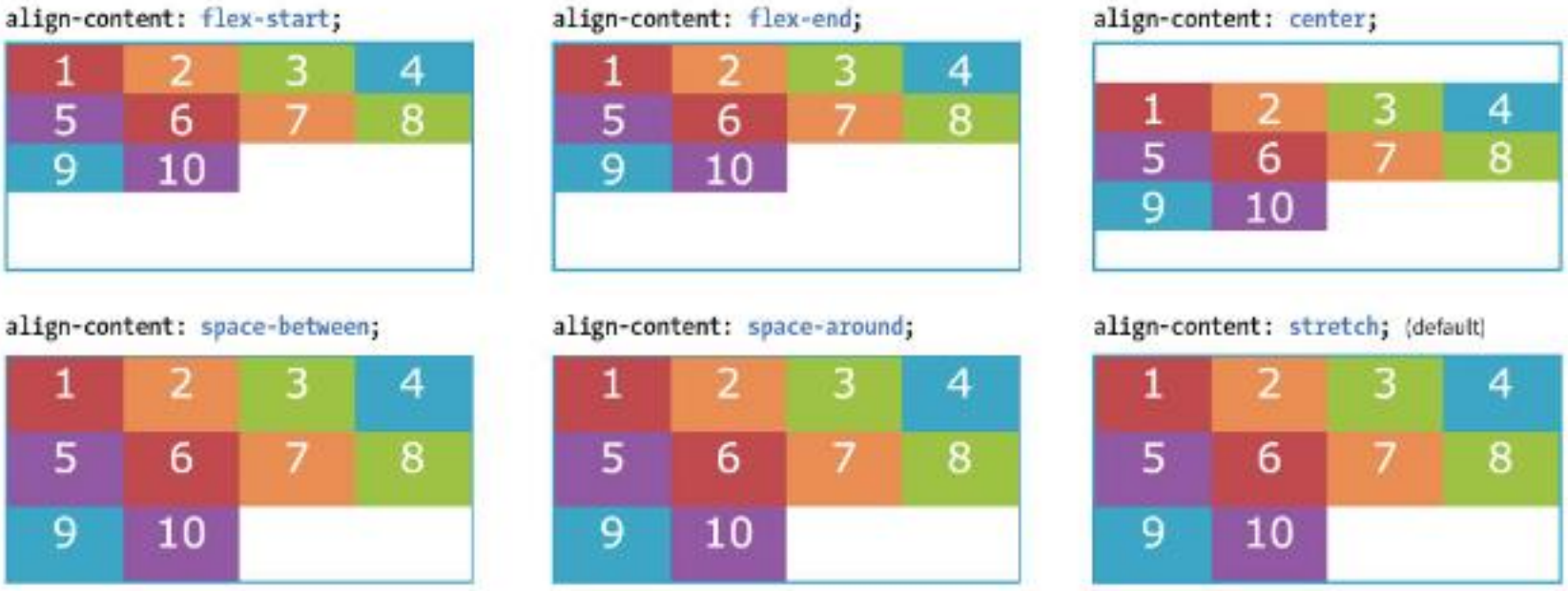


Figure 16-10. Niederst, J. (2018). Learning Web Design. O'Reilly Media, Inc.

The **align-content** property applies only when there are multiple wrapped flex lines

CSS Layout - Flexbox

Flex item - Properties

flex item properties determines how space is distributed ***within*** items

flex

None | *'flex-grow flex-shrink flex-basis'*

Flex: *flex-grow flex-shrink flex-basis*

Example

```
li {
   flex: 1 0 200px
}
```

Example

This list item starts at 200px wide
is allowed to grow to fill extra space
is NOT allowed to shrink below 200px

Values of 1 and 0 work as on/off switch
1 "turns on" or allows an item to grow or shrink
0 "turns off" prevents item from growing or shrinking

CSS Layout - Flexbox

Flex item - Properties

flex-grow

Value: *number* **Default:** *0*

`flex: 0 1 auto;` (prevents expansion)

`flex: 1 1 auto;` (allows expansion)

Figure 16-18. Niederst, J. (2018). Learning Web Design. O'Reilly Media, Inc.

THE MARKUP

```
<div id="container">
  <div class="box box1"> 1</div>
  <div class="box box2"> 2</div>
  <div class="box box3"> 3</div>
  <div class="box box4"> 4</div>
  <div class="box box5"> 5 </div>
</div>
```

THE STYLES

```
.box {
  …
  flex: 1 1 auto;
}
```

CSS Layout - Flexbox

Flex item - Properties

flex-grow

Value: *number* **Default:** *0*

Assigning a higher integer to an item – applies more space within that item.

```
.box4 {
   flex: 3 1 auto;
}
```

A flex-grow value of “3” means the item receives three times the amount of space than the remaining items set to flex-grow: 1

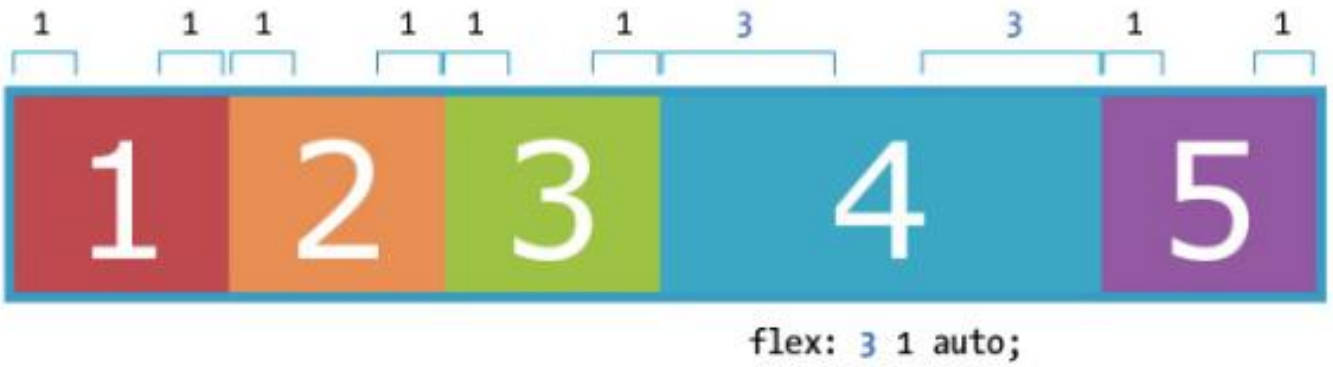


Figure 16-19. Niederst, J. (2018). Learning Web Design. O'Reilly Media, Inc.

CSS Layout - Flexbox

Flex item - Properties

flex-shrink

Value: *number* **Default:** *1*

When flex-shrink is 0, items are not permitted to shrink and may hang out of their containing element

Flex items stop shrinking when they reach their minimum size (defined by min-width/min-height).

```
box {
  flex: 0 1 100px;
}
```

flex: 0 1 100px;

When the container is wide, the items will not grow wider than their `flex-basis` of 100 pixels because `flex-grow` is set to 0.

When the container is narrow, the items are allowed to shrink to fit (`flex-shrink: 1`).

Figure 16-20. Niederst, J. (2018). Learning Web Design. O'Reilly Media, Inc.

CSS Layout - Flexbox

Flex item - Properties

flex-basis

Value: *length* | *percentage* | *content* | *auto*

```
box {
  flex: 0 1 100px;
}
```

Flex-basis defines the starting size of an item before any wrapping, growing or shrinking occurs. It may be used instead of the width property (or height for columns) for flex items.

`flex: 0 1 100px;`

When the container is wide, the items will not grow wider than their `flex-basis` of 100 pixels because `flex-grow` is set to 0.

When the container is narrow, the items are allowed to shrink to fit (`flex-shrink: 1`).

By default, flex-basis is set to auto (width/height value of item size)

If item's size is not defined or set to auto – flex-basis uses the content width.

Figure 16-20. Niederst, J. (2018). Learning Web Design. O'Reilly Media, Inc.

CSS Layout - Flexbox

Flex item - Properties

order

Value: *integer*

The order property give the ability to display items in an order that is different from the order in the HTML source code.

If items have the same order value = laid out in order they appear in code

If items have different order values = arranged from the lowest order value to highest.

```
.box3 {
  order: 1;
}
```

Figure 16-22. Niederst, J. (2018). Learning Web Design. O'Reilly Media, Inc.

CSS Layout - Flexbox

Flex item - Properties

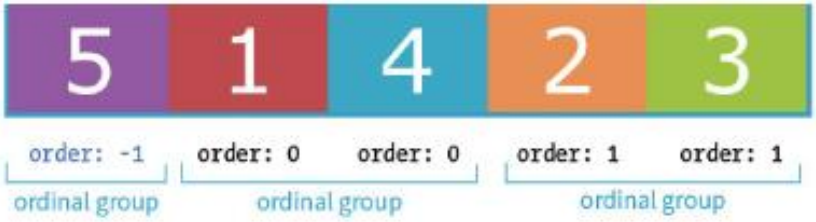
order

Value: *integer*

```
.box2, .box3 {
  order: 1
}
```

Figure 16-23&24. Niederst, J. (2018). Learning Web Design. O'Reilly Media, Inc.

```
.box5 {
  order: -1
}
```



CSS Layout - Flexbox

Flex item - Properties

order

Value: *integer*

```
main {
  display: flex;
}
article {
  flex: 1 1 50%;
  order: 2;
}
#news {
  flex: 1 1 25%;
  order: 3;
}
#contact {
  flex: 1 1 25%;
  order: 1;
}
```

Figure 16-25. Niederst, J. (2018). Learning Web Design. O'Reilly Media, Inc.

```
<header>…</header>
<main>
  <article><h2>Where It's At</h2></article>
  <aside id="news"><h2>News></h2></aside>
  <aside id="contact"><h2>Contact</h2><aside>
</main>
<footer>…<footer>
```

Let’s see how this works…