Introduction

Get started with Bootstrap, the world's most popular framework for building responsive, mobile-first sites, with BootstrapCDN and a template starter page.

Quick start

Looking to quickly add Bootstrap to your project? Use BootstrapCDN, provided for free by the folks at StackPath. Using a package manager or need to download the source files? Head to the downloads page.

CSS

Copy-paste the stylesheet `<link>` into your `<head>` before all other stylesheets to load our CSS.

Copy

```
<link rel="stylesheet" href="https://stackpath.bootstrapcdn.com/bootstrap/4.5.0/css/bootstrap.min.css" integrity="sha384-
9aIt2nRpC12Uk9gS9baDl411NQApFmC26EwAOH8WgZl5MYYxFfc+NcPb1dKGj7Sk" crossorigin="anonymous">
```

JS

Many of our components require the use of JavaScript to function. Specifically, they require jQuery, Popper.js, and our own JavaScript plugins. Place the following `<script>`s near the end of your pages, right before the closing `</body>` tag, to enable them. jQuery must come first, then Popper.js, and then our JavaScript plugins.

We use jQuery's slim build, but the full version is also supported.

Copy

```
<script src="https://code.jquery.com/jquery-3.5.1.slim.min.js" integrity="sha384-
DfXdz2htPH0lsSSs5nCTpuj/zy4C+OGpamoFVy38MVBnE+IbbVYUew+OrCXaRkfj" crossorigin="anonymous"></script>
<script src="https://cdn.jsdelivr.net/npm/popper.js@1.16.0/dist/umd/popper.min.js" integrity="sha384-
Q6E9RHvbIyZFJoft+2mJbHaEWldlvI9IOYy5n3zV9zzTtmI3UksdQRVvoxMfooAo" crossorigin="anonymous"></script>
<script src="https://stackpath.bootstrapcdn.com/bootstrap/4.5.0/js/bootstrap.min.js" integrity="sha384-
OgVRvuATP1z7JjHLkuOU7Xw704+h835Lr+6QL9UvYjZE3Ipu6Tp75j7Bh/kR0JKI" crossorigin="anonymous"></script>
```

Curious which components explicitly require jQuery, our JS, and Popper.js? Click the show components link below. If you're at all unsure about the general page structure, keep reading for an example page template.

Our `bootstrap.bundle.js` and `bootstrap.bundle.min.js` include Popper, but not jQuery. For more information about what's included in Bootstrap, please see our contents section.

▶ Show components requiring JavaScript

Starter template

Be sure to have your pages set up with the latest design and development standards. That means using an HTML5 doctype and including a viewport meta tag for proper responsive behaviors. Put it all together and your pages should look like this:

Copy

```html
<!doctype html>
<html lang="en">
  <head>
    <!-- Required meta tags -->
    <meta charset="utf-8">
    <meta name="viewport" content="width=device-width, initial-scale=1, shrink-to-fit=no">

    <!-- Bootstrap CSS -->
    <link rel="stylesheet" href="https://stackpath.bootstrapcdn.com/bootstrap/4.5.0/css/bootstrap.min.css" integrity="sha384-9aIt2nRpC12Uk9gS9baDl411NQApFmC26EwAOH8WgZl5MYYxFfc+NcPb1dKGj7Sk" crossorigin="anonymous">

    <title>Hello, world!</title>
  </head>
  <body>
    <h1>Hello, world!</h1>

    <!-- Optional JavaScript -->
    <!-- jQuery first, then Popper.js, then Bootstrap JS -->
    <script src="https://code.jquery.com/jquery-3.5.1.slim.min.js" integrity="sha384-DfXdz2htPH0lsSSs5nCTpuj/zy4C+OGpamoFVy38MVBnE+IbbVYUew+OrCXaRkfj" crossorigin="anonymous"></script>
    <script src="https://cdn.jsdelivr.net/npm/popper.js@1.16.0/dist/umd/popper.min.js" integrity="sha384-Q6E9RHvbIyZFJoft+2mJbHaEWldlvI9IOYy5n3zV9zzTtmI3UksdQRVvoxMfooAo" crossorigin="anonymous"></script>
    <script src="https://stackpath.bootstrapcdn.com/bootstrap/4.5.0/js/bootstrap.min.js" integrity="sha384-OgVRvuATP1z7JjHLkuOU7Xw704+h835Lr+6QL9UvYjZE3Ipu6Tp75j7Bh/kR0JKI" crossorigin="anonymous"></script>
  </body>
</html>
```

That's all you need for overall page requirements. Visit the Layout docs or our official examples to start laying out your site's content and components.

Important globals

Bootstrap employs a handful of important global styles and settings that you'll need to be aware of when using it, all of which are almost exclusively geared towards the *normalization* of cross browser styles. Let's dive in.

HTML5 doctype

Bootstrap requires the use of the HTML5 doctype. Without it, you'll see some funky incomplete styling, but including it shouldn't cause any considerable hiccups.

Copy

```html
<!doctype html>
<html lang="en">
  ...
</html>
```

Responsive meta tag

Bootstrap is developed *mobile first*, a strategy in which we optimize code for mobile devices first and then scale up components as necessary using CSS media queries. To ensure proper rendering and touch zooming for all devices, **add the responsive viewport meta tag** to your `<head>`.

Copy

```html
<meta name="viewport" content="width=device-width, initial-scale=1, shrink-to-fit=no">
```

You can see an example of this in action in the starter template.

Box-sizing

For more straightforward sizing in CSS, we switch the global `box-sizing` value from `content-box` to `border-box`. This ensures `padding` does not affect the final computed width of an element, but it can cause problems with some third party software like Google Maps and Google Custom Search Engine.

On the rare occasion you need to override it, use something like the following:

Copy

```
.selector-for-some-widget {
  box-sizing: content-box;
}
```

With the above snippet, nested elements—including generated content via `::before` and `::after`—will all inherit the specified `box-sizing` for that `.selector-for-some-widget`.

Learn more about box model and sizing at CSS Tricks.

Reboot

For improved cross-browser rendering, we use Reboot to correct inconsistencies across browsers and devices while providing slightly more opinionated resets to common HTML elements.

Community

Stay up to date on the development of Bootstrap and reach out to the community with these helpful resources.

- Follow @getbootstrap on Twitter.
- Read and subscribe to The Official Bootstrap Blog.
- Join the official Slack room.
- Chat with fellow Bootstrappers in IRC. On the `irc.freenode.net` server, in the `##bootstrap` channel.
- Implementation help may be found at Stack Overflow (tagged `bootstrap-4`).
- Developers should use the keyword `bootstrap` on packages which modify or add to the functionality of Bootstrap when distributing through npm or similar delivery mechanisms for maximum discoverability.

You can also follow @getbootstrap on Twitter for the latest gossip and awesome music videos.

Layout Overview

Components and options for laying out your Bootstrap project, including wrapping containers, a powerful grid system, a flexible media object, and responsive utility classes.

Containers

Containers are the most basic layout element in Bootstrap and are **required when using our default grid system**. Containers are used to contain, pad, and (sometimes) center the content within them. While containers *can* be nested, most layouts do not require a nested container.

Bootstrap comes with three different containers:

- `.container`, which sets a `max-width` at each responsive breakpoint
- `.container-fluid`, which is `width: 100%` at all breakpoints
- `.container-{breakpoint}`, which is `width: 100%` until the specified breakpoint

The table below illustrates how each container's `max-width` compares to the original `.container` and `.container-fluid` across each breakpoint.

See them in action and compare them in our Grid example.

	Extra small <576px	**Small** ≥576px	**Medium** ≥768px	**Large** ≥992px	**Extra large** ≥1200px
`.container`	100%	540px	720px	960px	1140px
`.container-sm`	100%	540px	720px	960px	1140px
`.container-md`	100%	100%	720px	960px	1140px
`.container-lg`	100%	100%	100%	960px	1140px
`.container-xl`	100%	100%	100%	100%	1140px
`.container-fluid`	100%	100%	100%	100%	100%

All-in-one

Our default `.container` class is a responsive, fixed-width container, meaning its `max-width` changes at each breakpoint.

Copy

```
<div class="container">
  <!-- Content here -->
</div>
```

Fluid

Use `.container-fluid` for a full width container, spanning the entire width of the viewport.

Copy

```
<div class="container-fluid">
  ...
</div>
```

Responsive

Responsive containers are new in Bootstrap v4.4. They allow you to specify a class that is 100% wide until the specified breakpoint is reached, after which we apply `max-width`s for each of the higher breakpoints. For example, `.container-sm` is 100% wide to start until the `sm` breakpoint is reached, where it will scale up with `md`, `lg`, and `xl`.

Copy

```
<div class="container-sm">100% wide until small breakpoint</div>
<div class="container-md">100% wide until medium breakpoint</div>
<div class="container-lg">100% wide until large breakpoint</div>
<div class="container-xl">100% wide until extra large breakpoint</div>
```

Responsive breakpoints

Since Bootstrap is developed to be mobile first, we use a handful of media queries to create sensible breakpoints for our layouts and interfaces. These breakpoints are mostly based on minimum viewport widths and allow us to scale up elements as the viewport changes.

Bootstrap primarily uses the following media query ranges—or breakpoints—in our source Sass files for our layout, grid system, and components.

Copy

```
// Extra small devices (portrait phones, less than 576px)
// No media query for `xs` since this is the default in Bootstrap

// Small devices (landscape phones, 576px and up)
@media (min-width: 576px) { ... }

// Medium devices (tablets, 768px and up)
@media (min-width: 768px) { ... }

// Large devices (desktops, 992px and up)
@media (min-width: 992px) { ... }

// Extra large devices (large desktops, 1200px and up)
@media (min-width: 1200px) { ... }
```

Since we write our source CSS in Sass, all our media queries are available via Sass mixins:

Copy

```
// No media query necessary for xs breakpoint as it's effectively `@media (min-width: 0) { ... }`
@include media-breakpoint-up(sm) { ... }
@include media-breakpoint-up(md) { ... }
@include media-breakpoint-up(lg) { ... }
@include media-breakpoint-up(xl) { ... }

// Example: Hide starting at `min-width: 0`, and then show at the `sm` breakpoint
.custom-class {
  display: none;
}
@include media-breakpoint-up(sm) {
  .custom-class {
    display: block;
  }
}
```

We occasionally use media queries that go in the other direction (the given screen size *or smaller*):

Copy

```
// Extra small devices (portrait phones, less than 576px)
@media (max-width: 575.98px) { ... }

// Small devices (landscape phones, less than 768px)
@media (max-width: 767.98px) { ... }

// Medium devices (tablets, less than 992px)
@media (max-width: 991.98px) { ... }

// Large devices (desktops, less than 1200px)
@media (max-width: 1199.98px) { ... }

// Extra large devices (large desktops)
// No media query since the extra-large breakpoint has no upper bound on its width
```

> Note that since browsers do not currently support range context queries, we work around the limitations of `min-` and `max-` prefixes and viewports with fractional widths (which can occur under certain conditions on high-dpi devices, for instance) by using values with higher precision for these comparisons.

Once again, these media queries are also available via Sass mixins:

Copy

```
@include media-breakpoint-down(xs) { ... }
@include media-breakpoint-down(sm) { ... }
@include media-breakpoint-down(md) { ... }
@include media-breakpoint-down(lg) { ... }
// No media query necessary for xl breakpoint as it has no upper bound on its width

// Example: Style from medium breakpoint and down
@include media-breakpoint-down(md) {
  .custom-class {
    display: block;
  }
}
```

There are also media queries and mixins for targeting a single segment of screen sizes using the minimum and maximum breakpoint widths.

Copy

```
// Extra small devices (portrait phones, less than 576px)
@media (max-width: 575.98px) { ... }

// Small devices (landscape phones, 576px and up)
@media (min-width: 576px) and (max-width: 767.98px) { ... }

// Medium devices (tablets, 768px and up)
@media (min-width: 768px) and (max-width: 991.98px) { ... }

// Large devices (desktops, 992px and up)
@media (min-width: 992px) and (max-width: 1199.98px) { ... }

// Extra large devices (large desktops, 1200px and up)
@media (min-width: 1200px) { ... }
```

These media queries are also available via Sass mixins:

Copy

```
@include media-breakpoint-only(xs) { ... }
@include media-breakpoint-only(sm) { ... }
@include media-breakpoint-only(md) { ... }
@include media-breakpoint-only(lg) { ... }
@include media-breakpoint-only(xl) { ... }
```

Similarly, media queries may span multiple breakpoint widths:

Copy

```
// Example
// Apply styles starting from medium devices and up to extra large devices
@media (min-width: 768px) and (max-width: 1199.98px) { ... }
```

The Sass mixin for targeting the same screen size range would be:

Copy

```
@include media-breakpoint-between(md, xl) { ... }
```

Z-index

Several Bootstrap components utilize `z-index`, the CSS property that helps control layout by providing a third axis to arrange content. We utilize a default z-index scale in Bootstrap that's been designed to properly layer navigation, tooltips and popovers, modals, and more.

These higher values start at an arbitrary number, high and specific enough to ideally avoid conflicts. We need a standard set of these across our layered components—tooltips, popovers, navbars, dropdowns, modals—so we can be reasonably consistent in the behaviors. There's no reason we couldn't have used `100`+ or `500`+.

We don't encourage customization of these individual values; should you change one, you likely need to change them all.

Copy

```
$zindex-dropdown:          1000 !default;
$zindex-sticky:            1020 !default;
$zindex-fixed:             1030 !default;
$zindex-modal-backdrop:    1040 !default;
$zindex-modal:             1050 !default;
$zindex-popover:           1060 !default;
$zindex-tooltip:           1070 !default;
```

To handle overlapping borders within components (e.g., buttons and inputs in input groups), we use low single digit `z-index` values of `1`, `2`, and `3` for default, hover, and active states. On hover/focus/active, we bring a particular element to the forefront with a higher `z-index` value to show their border over the sibling elements.

Utilities for layout

For faster mobile-friendly and responsive development, Bootstrap includes dozens of utility classes for showing, hiding, aligning, and spacing content.

Changing `display`

Use our display utilities for responsively toggling common values of the `display` property. Mix it with our grid system, content, or components to show or hide them across specific viewports.

Flexbox options

Bootstrap 4 is built with flexbox, but not every element's `display` has been changed to `display: flex` as this would add many unnecessary overrides and unexpectedly change key browser behaviors. Most of our components are built with flexbox enabled.

Should you need to add `display: flex` to an element, do so with `.d-flex` or one of the responsive variants (e.g., `.d-sm-flex`). You'll need this class or `display` value to allow the use of our extra flexbox utilities for sizing, alignment, spacing, and more.

Margin and padding

Use the `margin` and `padding` spacing utilities to control how elements and components are spaced and sized. Bootstrap 4 includes a five-level scale for spacing utilities, based on a `1rem` value default `$spacer` variable. Choose values for all viewports (e.g., `.mr-3` for `margin-right: 1rem`), or pick responsive variants to target specific viewports (e.g., `.mr-md-3` for `margin-right: 1rem` starting at the `md` breakpoint).

Toggle `visibility`

When toggling `display` isn't needed, you can toggle the `visibility` of an element with our visibility utilities. Invisible elements will still affect the layout of the page, but are visually hidden from visitors.

Flex

Quickly manage the layout, alignment, and sizing of grid columns, navigation, components, and more with a full suite of responsive flexbox utilities. For more complex implementations, custom CSS may be necessary.

Enable flex behaviors

Apply `display` utilities to create a flexbox container and transform **direct children elements** into flex items. Flex containers and items are able to be modified further with additional flex properties.

I'm a flexbox container!

Copy

```
<div class="d-flex p-2 bd-highlight">I'm a flexbox container!</div>
```

I'm an inline flexbox container!

Copy

```
<div class="d-inline-flex p-2 bd-highlight">I'm an inline flexbox container!</div>
```

Responsive variations also exist for `.d-flex` and `.d-inline-flex`.

- `.d-flex`
- `.d-inline-flex`
- `.d-sm-flex`
- `.d-sm-inline-flex`
- `.d-md-flex`
- `.d-md-inline-flex`
- `.d-lg-flex`
- `.d-lg-inline-flex`
- `.d-xl-flex`
- `.d-xl-inline-flex`

Direction

Set the direction of flex items in a flex container with direction utilities. In most cases you can omit the horizontal class here as the browser default is `row`. However, you may encounter situations where you needed to explicitly set this value (like responsive layouts).

Use `.flex-row` to set a horizontal direction (the browser default), or `.flex-row-reverse` to start the horizontal direction from the opposite side.

Flex item 1 Flex item 2 Flex item 3

Flex item 3 Flex item 2 Flex item 1

Copy

```html
<div class="d-flex flex-row bd-highlight mb-3">
  <div class="p-2 bd-highlight">Flex item 1</div>
  <div class="p-2 bd-highlight">Flex item 2</div>
  <div class="p-2 bd-highlight">Flex item 3</div>
</div>
<div class="d-flex flex-row-reverse bd-highlight">
  <div class="p-2 bd-highlight">Flex item 1</div>
  <div class="p-2 bd-highlight">Flex item 2</div>
  <div class="p-2 bd-highlight">Flex item 3</div>
</div>
```

Use `.flex-column` to set a vertical direction, or `.flex-column-reverse` to start the vertical direction from the opposite side.

Flex item 1
Flex item 2
Flex item 3

Flex item 3
Flex item 2
Flex item 1

Copy

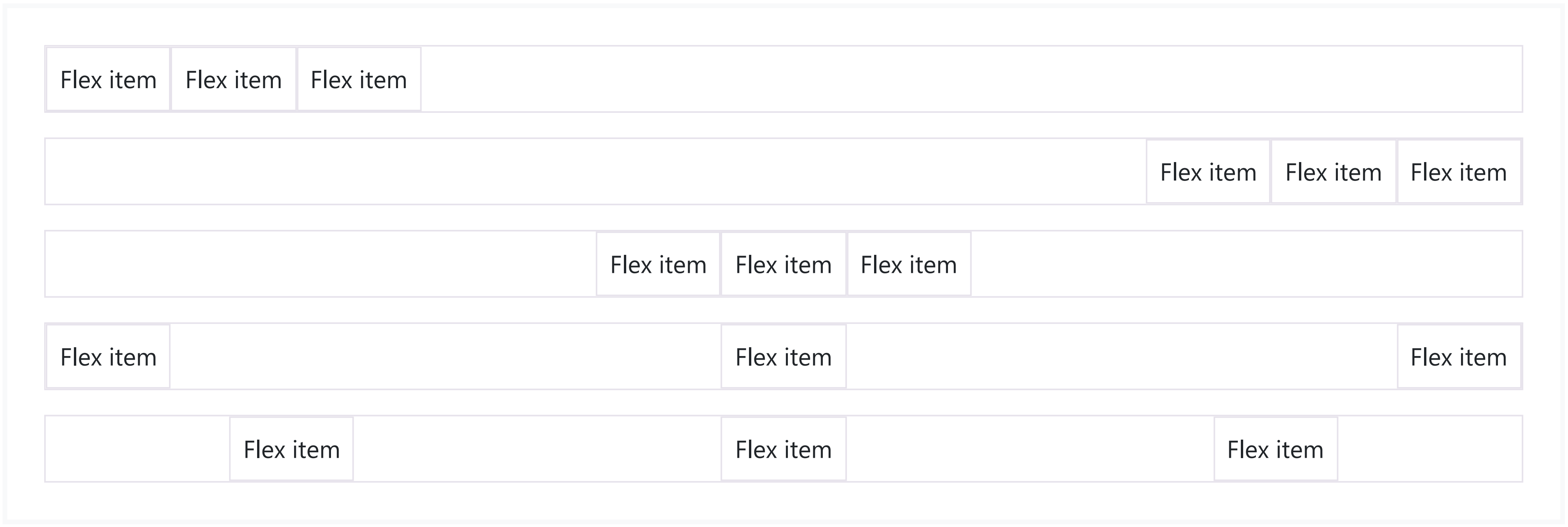
```html
<div class="d-flex flex-column bd-highlight mb-3">
  <div class="p-2 bd-highlight">Flex item 1</div>
  <div class="p-2 bd-highlight">Flex item 2</div>
  <div class="p-2 bd-highlight">Flex item 3</div>
</div>
<div class="d-flex flex-column-reverse bd-highlight">
  <div class="p-2 bd-highlight">Flex item 1</div>
  <div class="p-2 bd-highlight">Flex item 2</div>
  <div class="p-2 bd-highlight">Flex item 3</div>
</div>
```

Responsive variations also exist for `flex-direction`.

- `.flex-row`
- `.flex-row-reverse`
- `.flex-column`
- `.flex-column-reverse`
- `.flex-sm-row`
- `.flex-sm-row-reverse`
- `.flex-sm-column`
- `.flex-sm-column-reverse`
- `.flex-md-row`
- `.flex-md-row-reverse`
- `.flex-md-column`
- `.flex-md-column-reverse`
- `.flex-lg-row`
- `.flex-lg-row-reverse`
- `.flex-lg-column`
- `.flex-lg-column-reverse`
- `.flex-xl-row`
- `.flex-xl-row-reverse`
- `.flex-xl-column`
- `.flex-xl-column-reverse`

Justify content

Use `justify-content` utilities on flexbox containers to change the alignment of flex items on the main axis (the x-axis to start, y-axis if `flex-direction: column`). Choose from `start` (browser default), `end`, `center`, `between`, or `around`.



Copy

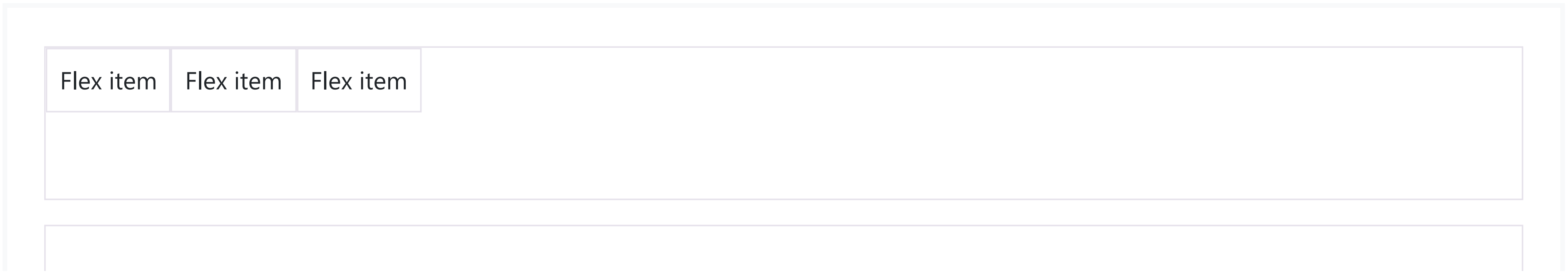
```html
<div class="d-flex justify-content-start">...</div>
<div class="d-flex justify-content-end">...</div>
<div class="d-flex justify-content-center">...</div>
<div class="d-flex justify-content-between">...</div>
<div class="d-flex justify-content-around">...</div>
```

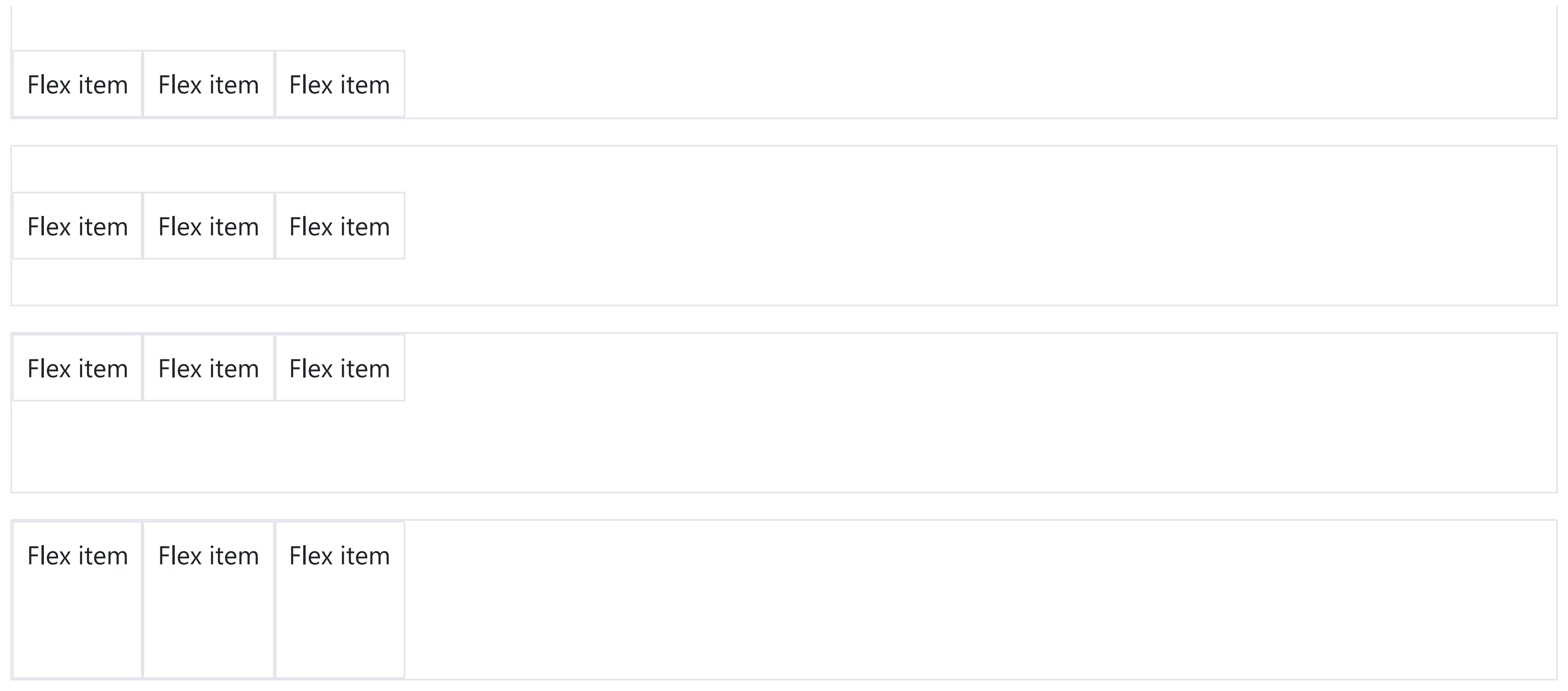
Responsive variations also exist for `justify-content`.

- `.justify-content-start`
- `.justify-content-end`
- `.justify-content-center`
- `.justify-content-between`
- `.justify-content-around`
- `.justify-content-sm-start`
- `.justify-content-sm-end`
- `.justify-content-sm-center`
- `.justify-content-sm-between`
- `.justify-content-sm-around`
- `.justify-content-md-start`
- `.justify-content-md-end`
- `.justify-content-md-center`
- `.justify-content-md-between`
- `.justify-content-md-around`
- `.justify-content-lg-start`
- `.justify-content-lg-end`
- `.justify-content-lg-center`
- `.justify-content-lg-between`
- `.justify-content-lg-around`
- `.justify-content-xl-start`
- `.justify-content-xl-end`
- `.justify-content-xl-center`
- `.justify-content-xl-between`
- `.justify-content-xl-around`

Align items

Use `align-items` utilities on flexbox containers to change the alignment of flex items on the cross axis (the y-axis to start, x-axis if `flex-direction: column`). Choose from `start`, `end`, `center`, `baseline`, or `stretch` (browser default).





Copy

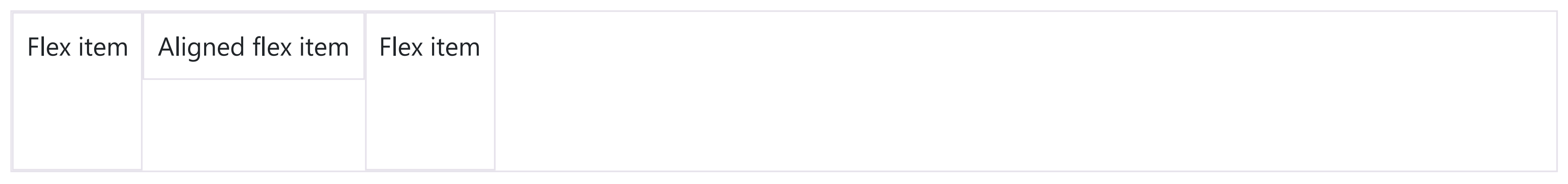
```html
<div class="d-flex align-items-start">...</div>
<div class="d-flex align-items-end">...</div>
<div class="d-flex align-items-center">...</div>
<div class="d-flex align-items-baseline">...</div>
<div class="d-flex align-items-stretch">...</div>
```

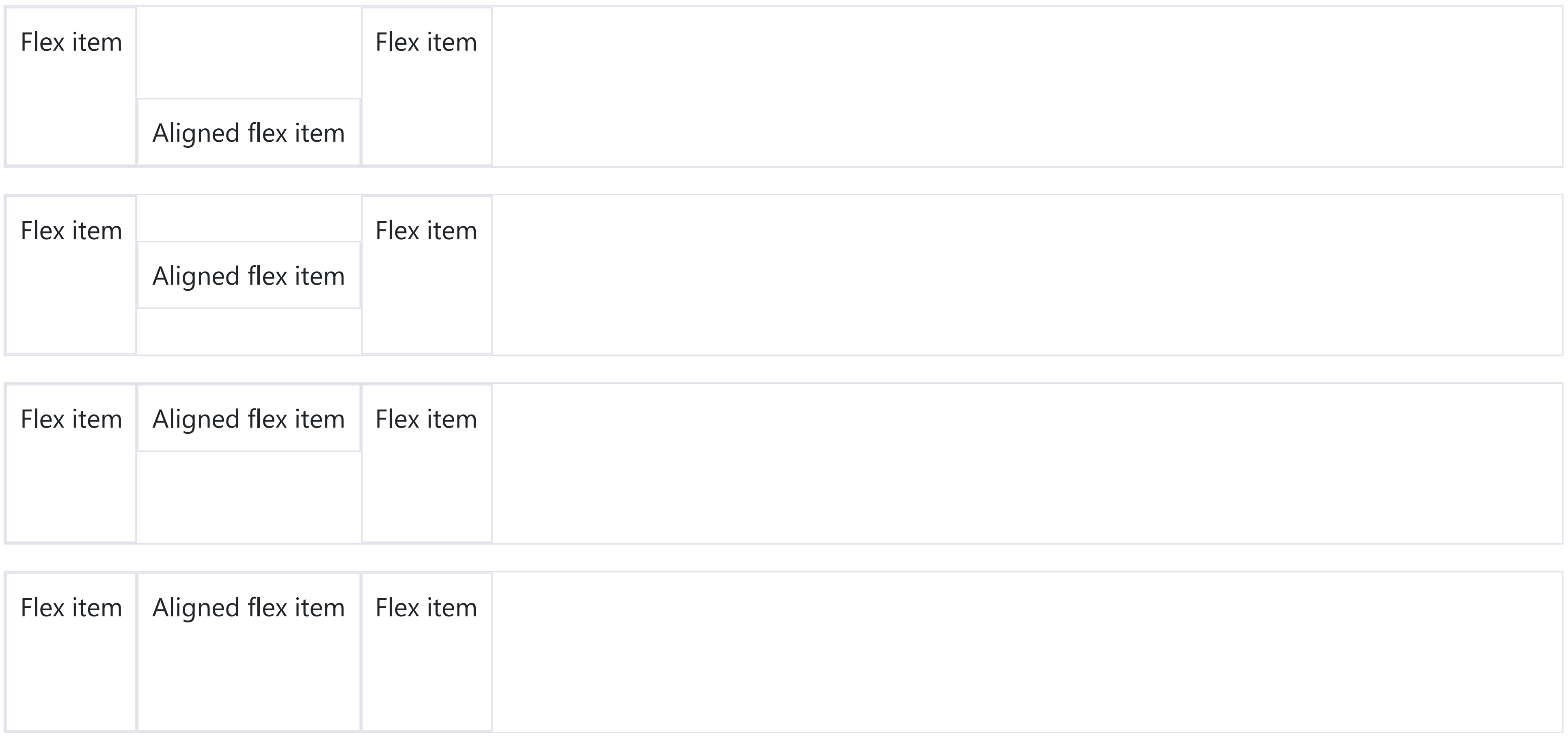
Responsive variations also exist for `align-items`.

- `.align-items-start`
- `.align-items-end`
- `.align-items-center`
- `.align-items-baseline`
- `.align-items-stretch`
- `.align-items-sm-start`
- `.align-items-sm-end`
- `.align-items-sm-center`
- `.align-items-sm-baseline`
- `.align-items-sm-stretch`
- `.align-items-md-start`
- `.align-items-md-end`
- `.align-items-md-center`
- `.align-items-md-baseline`
- `.align-items-md-stretch`
- `.align-items-lg-start`
- `.align-items-lg-end`
- `.align-items-lg-center`
- `.align-items-lg-baseline`
- `.align-items-lg-stretch`
- `.align-items-xl-start`
- `.align-items-xl-end`
- `.align-items-xl-center`
- `.align-items-xl-baseline`
- `.align-items-xl-stretch`

Align self

Use `align-self` utilities on flexbox items to individually change their alignment on the cross axis (the y-axis to start, x-axis if `flex-direction: column`). Choose from the same options as `align-items`: `start`, `end`, `center`, `baseline`, or `stretch` (browser default).





Copy

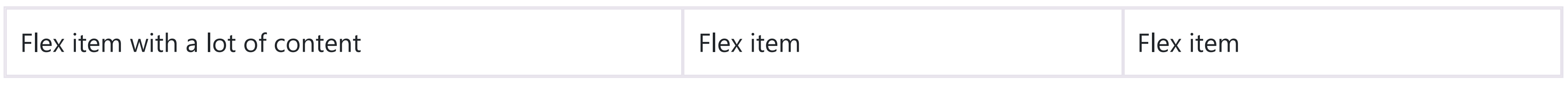
```
<div class="align-self-start">Aligned flex item</div>
<div class="align-self-end">Aligned flex item</div>
<div class="align-self-center">Aligned flex item</div>
<div class="align-self-baseline">Aligned flex item</div>
<div class="align-self-stretch">Aligned flex item</div>
```

Responsive variations also exist for `align-self`.

- `.align-self-start`
- `.align-self-end`
- `.align-self-center`
- `.align-self-baseline`
- `.align-self-stretch`
- `.align-self-sm-start`
- `.align-self-sm-end`
- `.align-self-sm-center`
- `.align-self-sm-baseline`
- `.align-self-sm-stretch`
- `.align-self-md-start`
- `.align-self-md-end`
- `.align-self-md-center`
- `.align-self-md-baseline`
- `.align-self-md-stretch`
- `.align-self-lg-start`
- `.align-self-lg-end`
- `.align-self-lg-center`
- `.align-self-lg-baseline`
- `.align-self-lg-stretch`
- `.align-self-xl-start`
- `.align-self-xl-end`
- `.align-self-xl-center`
- `.align-self-xl-baseline`
- `.align-self-xl-stretch`

Fill

Use the `.flex-fill` class on a series of sibling elements to force them into widths equal to their content (or equal widths if their content does not surpass their border-boxes) while taking up all available horizontal space.



Copy

```html
<div class="d-flex bd-highlight">
  <div class="p-2 flex-fill bd-highlight">Flex item with a lot of content</div>
  <div class="p-2 flex-fill bd-highlight">Flex item</div>
  <div class="p-2 flex-fill bd-highlight">Flex item</div>
</div>
```

Responsive variations also exist for `flex-fill`.

- `.flex-fill`
- `.flex-sm-fill`
- `.flex-md-fill`
- `.flex-lg-fill`
- `.flex-xl-fill`

Grow and shrink

Use `.flex-grow-*` utilities to toggle a flex item's ability to grow to fill available space. In the example below, the `.flex-grow-1` elements uses all available space it can, while allowing the remaining two flex items their necessary space.

Flex item | Flex item | Third flex item

Copy

```html
<div class="d-flex bd-highlight">
  <div class="p-2 flex-grow-1 bd-highlight">Flex item</div>
  <div class="p-2 bd-highlight">Flex item</div>
  <div class="p-2 bd-highlight">Third flex item</div>
</div>
```

Use `.flex-shrink-*` utilities to toggle a flex item's ability to shrink if necessary. In the example below, the second flex item with `.flex-shrink-1` is forced to wrap its contents to a new line, "shrinking" to allow more space for the previous flex item with `.w-100`.

Flex item | Flex item

Copy

```html
<div class="d-flex bd-highlight">
  <div class="p-2 w-100 bd-highlight">Flex item</div>
  <div class="p-2 flex-shrink-1 bd-highlight">Flex item</div>
</div>
```

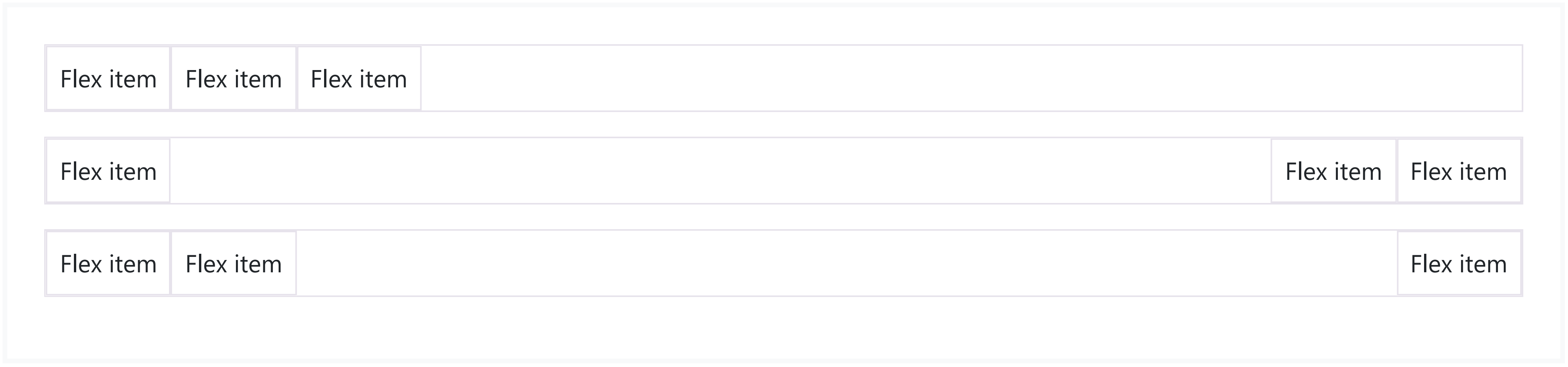
Responsive variations also exist for `flex-grow` and `flex-shrink`.

- `.flex-{grow|shrink}-0`
- `.flex-{grow|shrink}-1`
- `.flex-sm-{grow|shrink}-0`
- `.flex-sm-{grow|shrink}-1`
- `.flex-md-{grow|shrink}-0`
- `.flex-md-{grow|shrink}-1`
- `.flex-lg-{grow|shrink}-0`
- `.flex-lg-{grow|shrink}-1`
- `.flex-xl-{grow|shrink}-0`
- `.flex-xl-{grow|shrink}-1`

Auto margins

Flexbox can do some pretty awesome things when you mix flex alignments with auto margins. Shown below are three examples of controlling flex items via auto margins: default (no auto margin), pushing two items to the right (`.mr-auto`), and pushing two items to the left (`.ml-auto`).

Unfortunately, IE10 and IE11 do not properly support auto margins on flex items whose parent has a non-default `justify-content` value. See this StackOverflow answer for more details.



Copy

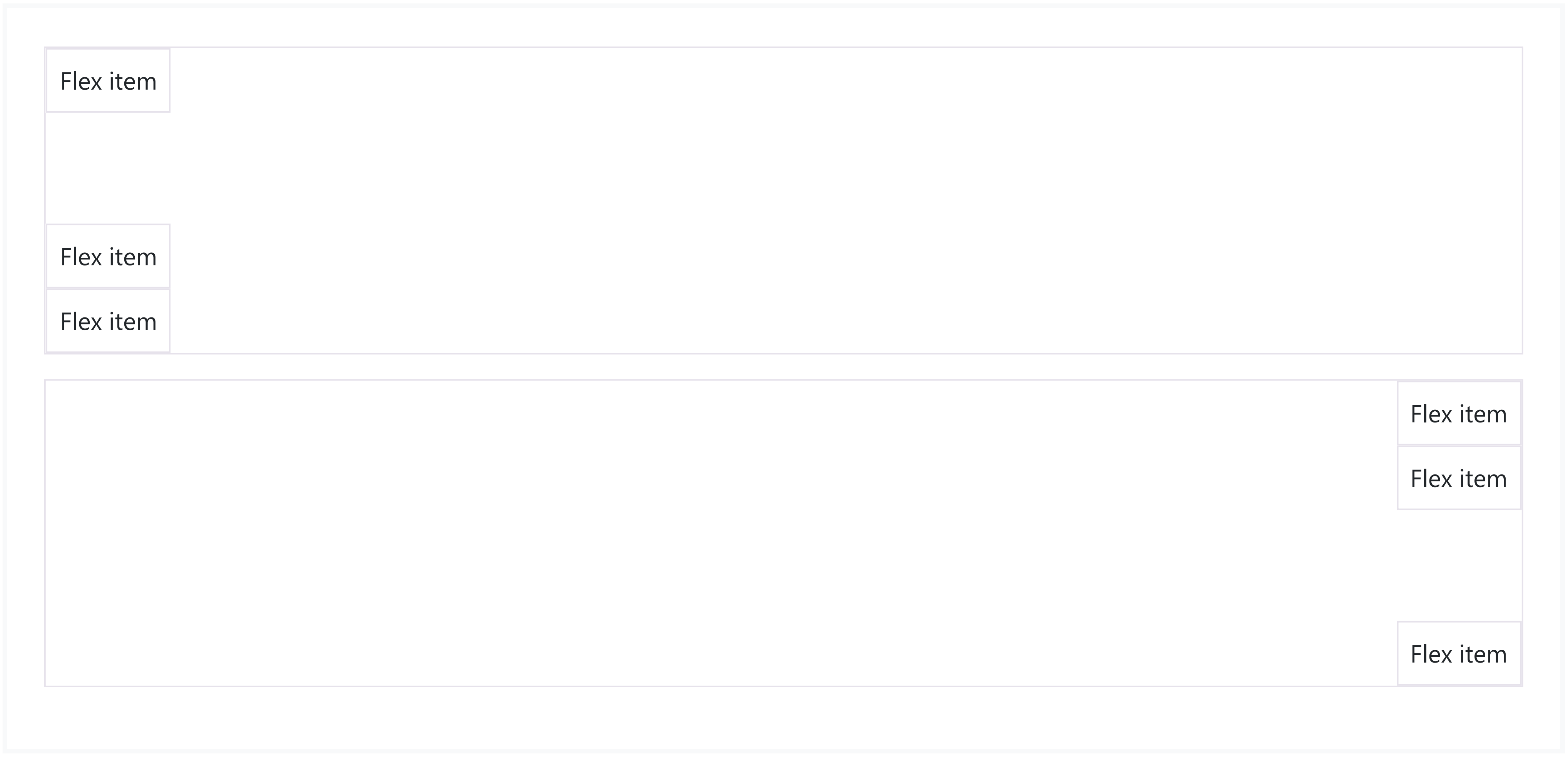
```
<div class="d-flex bd-highlight mb-3">
  <div class="p-2 bd-highlight">Flex item</div>
  <div class="p-2 bd-highlight">Flex item</div>
  <div class="p-2 bd-highlight">Flex item</div>
</div>

<div class="d-flex bd-highlight mb-3">
  <div class="mr-auto p-2 bd-highlight">Flex item</div>
  <div class="p-2 bd-highlight">Flex item</div>
  <div class="p-2 bd-highlight">Flex item</div>
</div>

<div class="d-flex bd-highlight mb-3">
  <div class="p-2 bd-highlight">Flex item</div>
  <div class="p-2 bd-highlight">Flex item</div>
  <div class="ml-auto p-2 bd-highlight">Flex item</div>
</div>
```

With align-items

Vertically move one flex item to the top or bottom of a container by mixing `align-items`, `flex-direction: column`, and `margin-top: auto` or `margin-bottom: auto`.



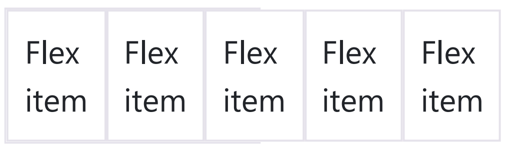
Copy

```
<div class="d-flex align-items-start flex-column bd-highlight mb-3" style="height: 200px;">
  <div class="mb-auto p-2 bd-highlight">Flex item</div>
  <div class="p-2 bd-highlight">Flex item</div>
  <div class="p-2 bd-highlight">Flex item</div>
</div>

<div class="d-flex align-items-end flex-column bd-highlight mb-3" style="height: 200px;">
  <div class="p-2 bd-highlight">Flex item</div>
  <div class="p-2 bd-highlight">Flex item</div>
  <div class="mt-auto p-2 bd-highlight">Flex item</div>
</div>
```

Wrap

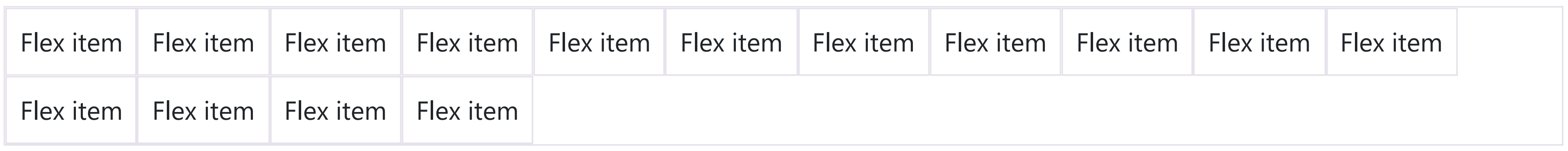
Change how flex items wrap in a flex container. Choose from no wrapping at all (the browser default) with `.flex-nowrap`, wrapping with `.flex-wrap`, or reverse wrapping with `.flex-wrap-reverse`.



Flex item Flex item Flex item Flex item Flex item

Copy

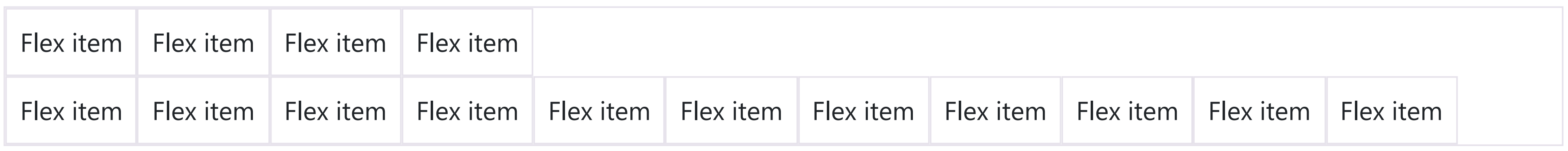
```
<div class="d-flex flex-nowrap">
  ...
</div>
```



Flex item Flex item Flex item Flex item Flex item Flex item Flex item Flex item Flex item Flex item Flex item
Flex item Flex item Flex item Flex item

Copy

```
<div class="d-flex flex-wrap">
  ...
</div>
```



Flex item Flex item Flex item Flex item
Flex item Flex item Flex item Flex item Flex item Flex item Flex item Flex item Flex item Flex item Flex item

Copy

```
<div class="d-flex flex-wrap-reverse">
  ...
</div>
```

Responsive variations also exist for `flex-wrap`.

- `.flex-nowrap`
- `.flex-wrap`
- `.flex-wrap-reverse`
- `.flex-sm-nowrap`
- `.flex-sm-wrap`
- `.flex-sm-wrap-reverse`
- `.flex-md-nowrap`
- `.flex-md-wrap`
- `.flex-md-wrap-reverse`
- `.flex-lg-nowrap`
- `.flex-lg-wrap`
- `.flex-lg-wrap-reverse`
- `.flex-xl-nowrap`
- `.flex-xl-wrap`
- `.flex-xl-wrap-reverse`

Order

Change the *visual* order of specific flex items with a handful of `order` utilities. We only provide options for making an item first or last, as well as a reset to use the DOM order. As `order` takes any integer value (e.g., `5`), add custom CSS for any additional values needed.

Third flex item Second flex item First flex item

Copy

```
<div class="d-flex flex-nowrap bd-highlight">
  <div class="order-3 p-2 bd-highlight">First flex item</div>
  <div class="order-2 p-2 bd-highlight">Second flex item</div>
  <div class="order-1 p-2 bd-highlight">Third flex item</div>
</div>
```

Responsive variations also exist for `order`.

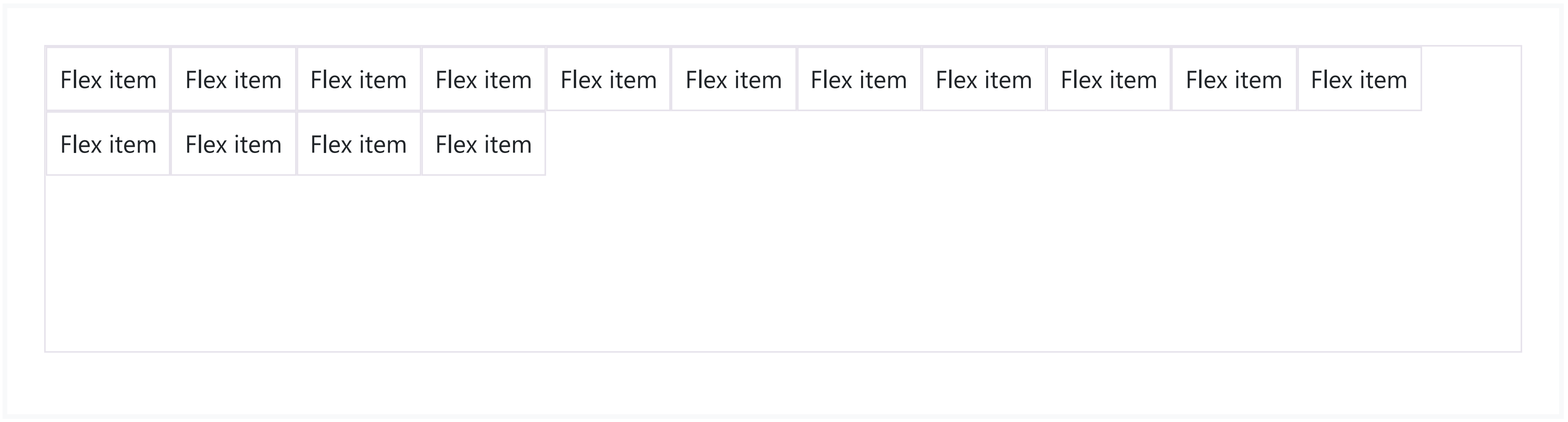
- `.order-0`
- `.order-1`
- `.order-2`
- `.order-3`
- `.order-4`
- `.order-5`
- `.order-6`
- `.order-7`
- `.order-8`
- `.order-9`
- `.order-10`
- `.order-11`
- `.order-12`
- `.order-sm-0`
- `.order-sm-1`
- `.order-sm-2`
- `.order-sm-3`
- `.order-sm-4`
- `.order-sm-5`
- `.order-sm-6`
- `.order-sm-7`
- `.order-sm-8`
- `.order-sm-9`
- `.order-sm-10`
- `.order-sm-11`
- `.order-sm-12`
- `.order-md-0`
- `.order-md-1`
- `.order-md-2`
- `.order-md-3`
- `.order-md-4`
- `.order-md-5`
- `.order-md-6`
- `.order-md-7`
- `.order-md-8`
- `.order-md-9`
- `.order-md-10`
- `.order-md-11`
- `.order-md-12`
- `.order-lg-0`
- `.order-lg-1`
- `.order-lg-2`
- `.order-lg-3`
- `.order-lg-4`
- `.order-lg-5`
- `.order-lg-6`
- `.order-lg-7`
- `.order-lg-8`
- `.order-lg-9`
- `.order-lg-10`
- `.order-lg-11`

- `.order-lg-12`
- `.order-xl-0`
- `.order-xl-1`
- `.order-xl-2`
- `.order-xl-3`
- `.order-xl-4`
- `.order-xl-5`
- `.order-xl-6`
- `.order-xl-7`
- `.order-xl-8`
- `.order-xl-9`
- `.order-xl-10`
- `.order-xl-11`
- `.order-xl-12`

Align content

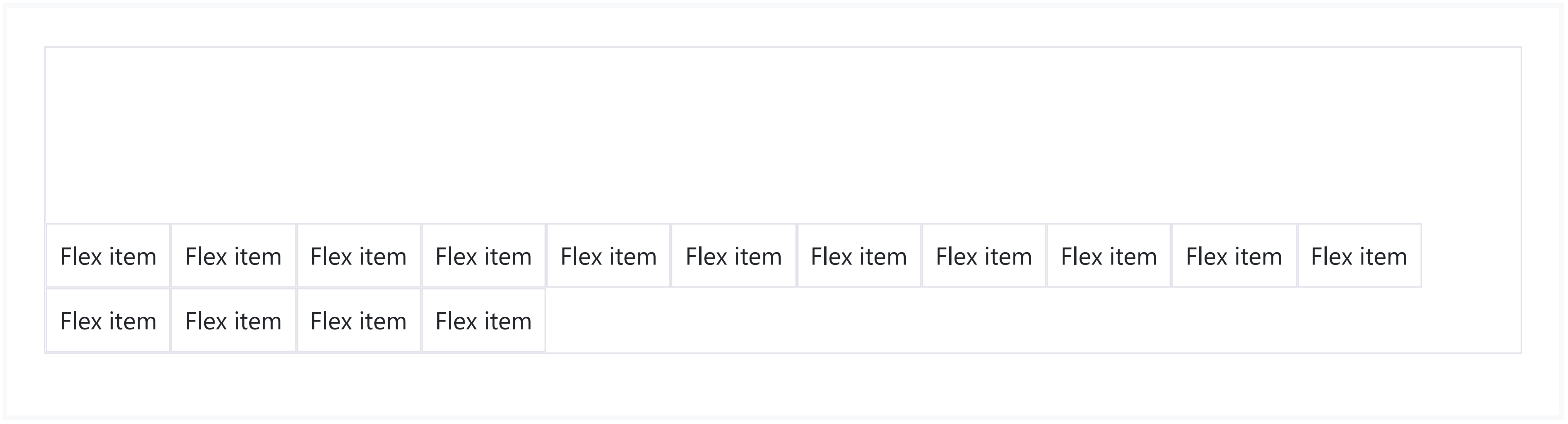
Use `align-content` utilities on flexbox containers to align flex items *together* on the cross axis. Choose from `start` (browser default), `end`, `center`, `between`, `around`, or `stretch`. To demonstrate these utilities, we've enforced `flex-wrap: wrap` and increased the number of flex items.

Heads up! This property has no effect on single rows of flex items.



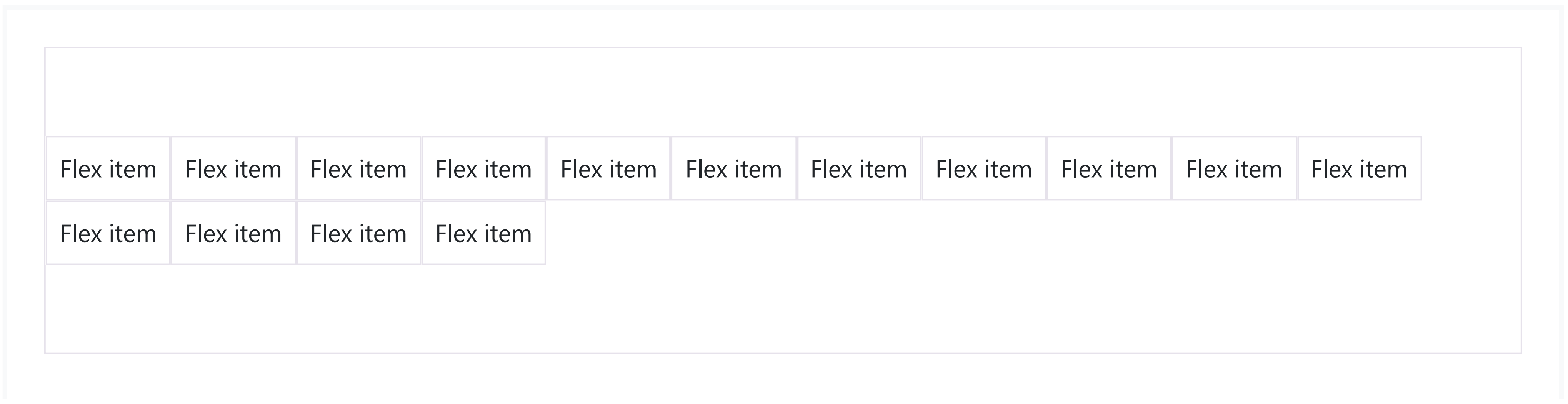
Copy

```
<div class="d-flex align-content-start flex-wrap">
  ...
</div>
```



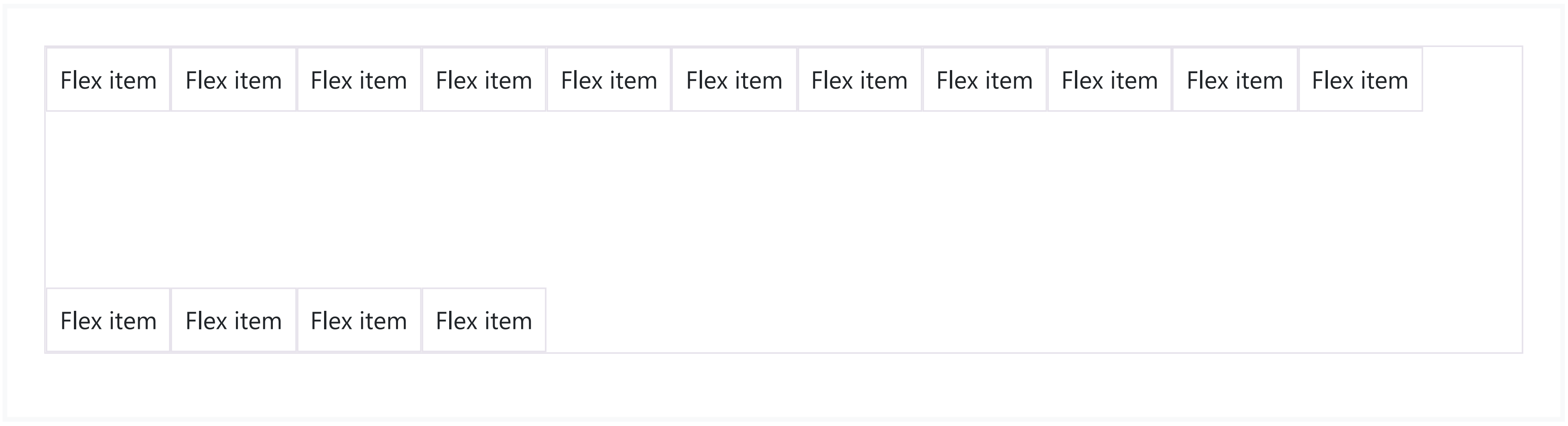
Copy

```
<div class="d-flex align-content-end flex-wrap">...</div>
```



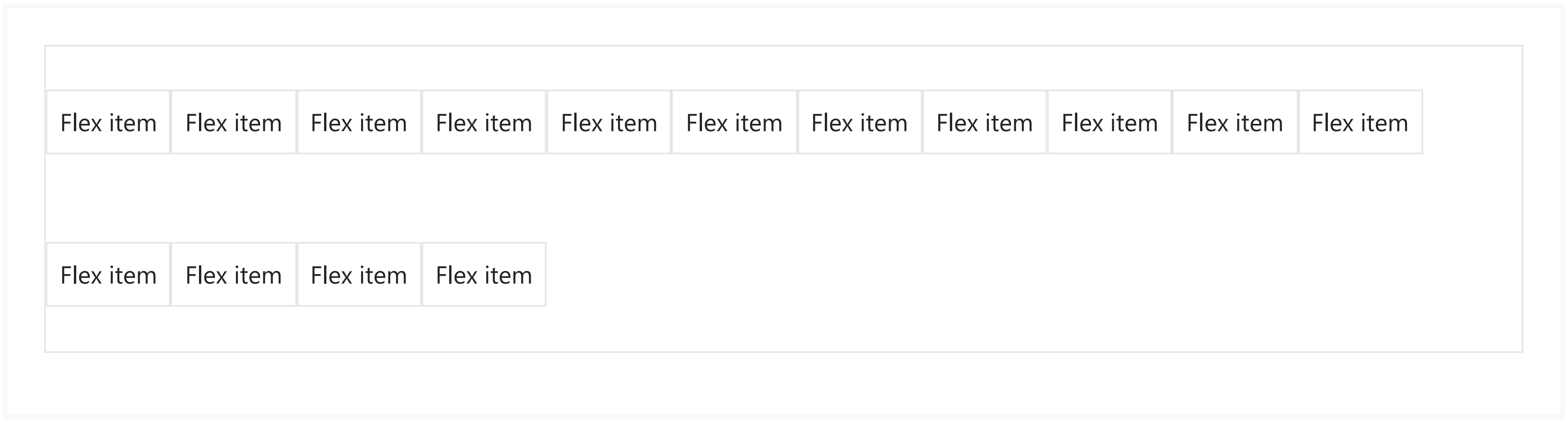
Copy

```
<div class="d-flex align-content-center flex-wrap">...</div>
```



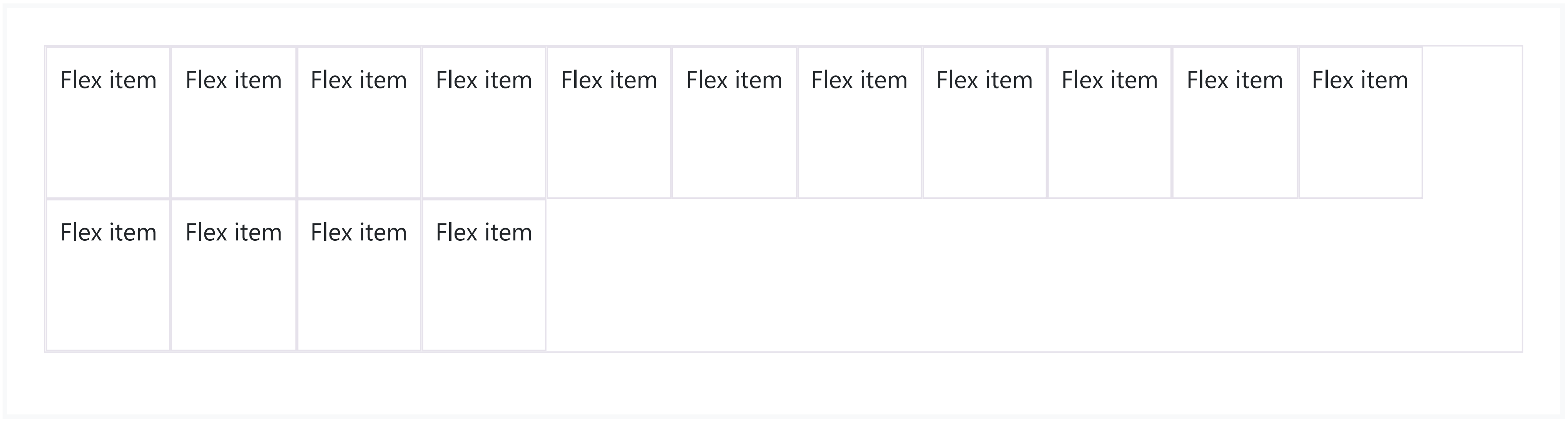
Copy

```
<div class="d-flex align-content-between flex-wrap">...</div>
```



Copy

```
<div class="d-flex align-content-around flex-wrap">...</div>
```



Copy

```
<div class="d-flex align-content-stretch flex-wrap">...</div>
```

Responsive variations also exist for `align-content`.

- `.align-content-start`
- `.align-content-end`
- `.align-content-center`
- `.align-content-around`
- `.align-content-stretch`
- `.align-content-sm-start`
- `.align-content-sm-end`
- `.align-content-sm-center`
- `.align-content-sm-around`
- `.align-content-sm-stretch`
- `.align-content-md-start`
- `.align-content-md-end`

- `.align-content-md-center`
- `.align-content-md-around`
- `.align-content-md-stretch`
- `.align-content-lg-start`
- `.align-content-lg-end`
- `.align-content-lg-center`
- `.align-content-lg-around`
- `.align-content-lg-stretch`
- `.align-content-xl-start`
- `.align-content-xl-end`
- `.align-content-xl-center`
- `.align-content-xl-around`
- `.align-content-xl-stretch`

Grid system

Use our powerful mobile-first flexbox grid to build layouts of all shapes and sizes thanks to a twelve column system, five default responsive tiers, Sass variables and mixins, and dozens of predefined classes.

How it works

Bootstrap's grid system uses a series of containers, rows, and columns to layout and align content. It's built with flexbox and is fully responsive. Below is an example and an in-depth look at how the grid comes together.

New to or unfamiliar with flexbox? Read this CSS Tricks flexbox guide for background, terminology, guidelines, and code snippets.

One of three columns	One of three columns	One of three columns

Copy

```
<div class="container">
  <div class="row">
    <div class="col-sm">
      One of three columns
    </div>
    <div class="col-sm">
      One of three columns
    </div>
    <div class="col-sm">
      One of three columns
    </div>
  </div>
</div>
```

The above example creates three equal-width columns on small, medium, large, and extra large devices using our predefined grid classes. Those columns are centered in the page with the parent `.container`.

Breaking it down, here's how it works:

- Containers provide a means to center and horizontally pad your site's contents. Use `.container` for a responsive pixel width or `.container-fluid` for `width: 100%` across all viewport and device sizes.
- Rows are wrappers for columns. Each column has horizontal `padding` (called a gutter) for controlling the space between them. This `padding` is then counteracted on the rows with negative margins. This way, all the content in your columns is visually aligned down the left side.
- In a grid layout, content must be placed within columns and only columns may be immediate children of rows.
- Thanks to flexbox, grid columns without a specified `width` will automatically layout as equal width columns. For example, four instances of `.col-sm` will each automatically be 25% wide from the small breakpoint and up. See the auto-layout columns section for more examples.
- Column classes indicate the number of columns you'd like to use out of the possible 12 per row. So, if you want three equal-width columns across, you can use `.col-4`.
- Column `width`s are set in percentages, so they're always fluid and sized relative to their parent element.
- Columns have horizontal `padding` to create the gutters between individual columns, however, you can remove the `margin` from rows and `padding` from columns with `.no-gutters` on the `.row`.
- To make the grid responsive, there are five grid breakpoints, one for each responsive breakpoint: all breakpoints (extra small), small, medium, large, and extra large.
- Grid breakpoints are based on minimum width media queries, meaning **they apply to that one breakpoint and all those above it** (e.g., `.col-sm-4` applies to small, medium, large, and extra large devices, but not the first `xs` breakpoint).
- You can use predefined grid classes (like `.col-4`) or Sass mixins for more semantic markup.

Be aware of the limitations and bugs around flexbox, like the inability to use some HTML elements as flex containers.

Grid options

While Bootstrap uses `em`s or `rem`s for defining most sizes, `px`s are used for grid breakpoints and container widths. This is because the viewport width is in pixels and does not change with the font size.

See how aspects of the Bootstrap grid system work across multiple devices with a handy table.

	Extra small <576px	**Small** ≥576px	**Medium** ≥768px	**Large** ≥992px	**Extra large** ≥1200px
Max container width	None (auto)	540px	720px	960px	1140px
Class prefix	`.col-`	`.col-sm-`	`.col-md-`	`.col-lg-`	`.col-xl-`
# of columns	12				
Gutter width	30px (15px on each side of a column)				
Nestable	Yes				
Column ordering	Yes				

Auto-layout columns

Utilize breakpoint-specific column classes for easy column sizing without an explicit numbered class like `.col-sm-6`.

Equal-width

For example, here are two grid layouts that apply to every device and viewport, from `xs` to `xl`. Add any number of unit-less classes for each breakpoint you need and every column will be the same width.

1 of 2	2 of 2

1 of 3	2 of 3	3 of 3

Copy

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

Equal-width multi-line

Create equal-width columns that span multiple lines by inserting a `.w-100` where you want the columns to break to a new line. Make the breaks responsive by mixing `.w-100` with some responsive display utilities.

There was a Safari flexbox bug that prevented this from working without an explicit `flex-basis` or `border`. There are workarounds for older browser versions, but they shouldn't be necessary if your target browsers don't fall into the buggy versions.

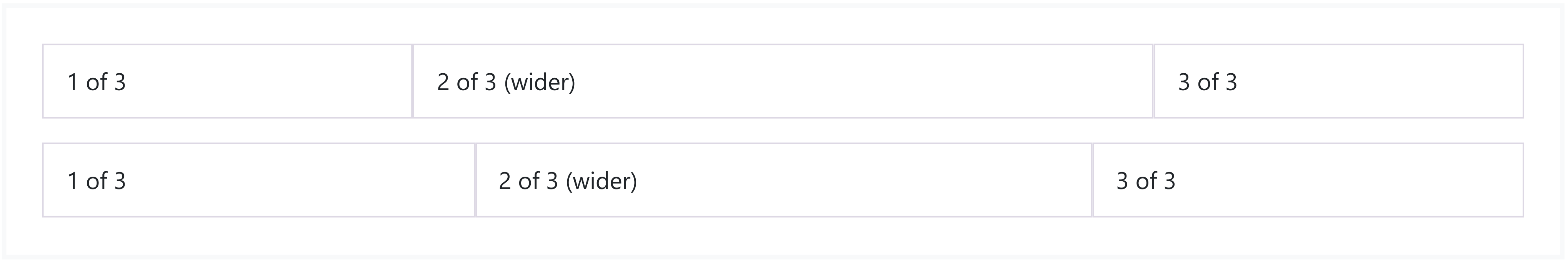
col	col
col	col

Copy

```
<div class="container">
  <div class="row">
    <div class="col">col</div>
    <div class="col">col</div>
    <div class="w-100"></div>
    <div class="col">col</div>
    <div class="col">col</div>
  </div>
</div>
```

Setting one column width

Auto-layout for flexbox grid columns also means you can set the width of one column and have the sibling columns automatically resize around it. You may use predefined grid classes (as shown below), grid mixins, or inline widths. Note that the other columns will resize no matter the width of the center column.


1 of 3 | 2 of 3 (wider) | 3 of 3

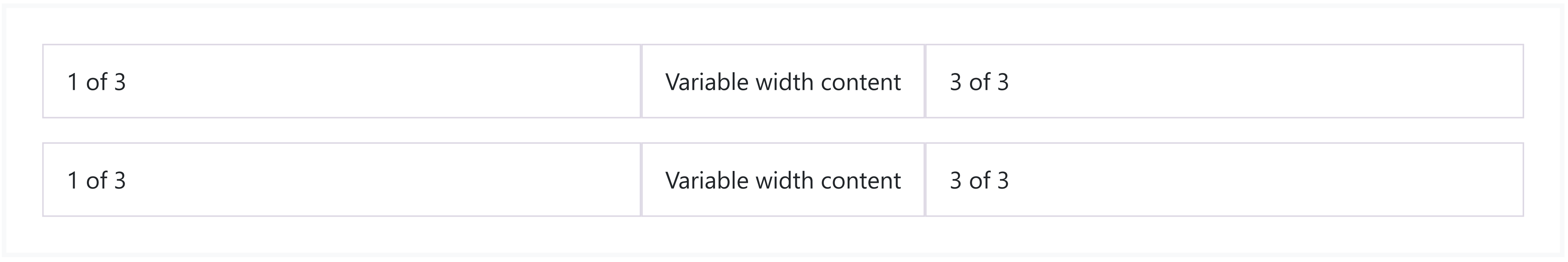
1 of 3 | 2 of 3 (wider) | 3 of 3

Copy

```
<div class="container">
  <div class="row">
    <div class="col">
      1 of 3
    </div>
    <div class="col-6">
      2 of 3 (wider)
    </div>
    <div class="col">
      3 of 3
    </div>
  </div>
  <div class="row">
    <div class="col">
      1 of 3
    </div>
    <div class="col-5">
      2 of 3 (wider)
    </div>
    <div class="col">
      3 of 3
    </div>
  </div>
</div>
```

Variable width content

Use `col-{breakpoint}-auto` classes to size columns based on the natural width of their content.


1 of 3 | Variable width content | 3 of 3

1 of 3 | Variable width content | 3 of 3

Copy

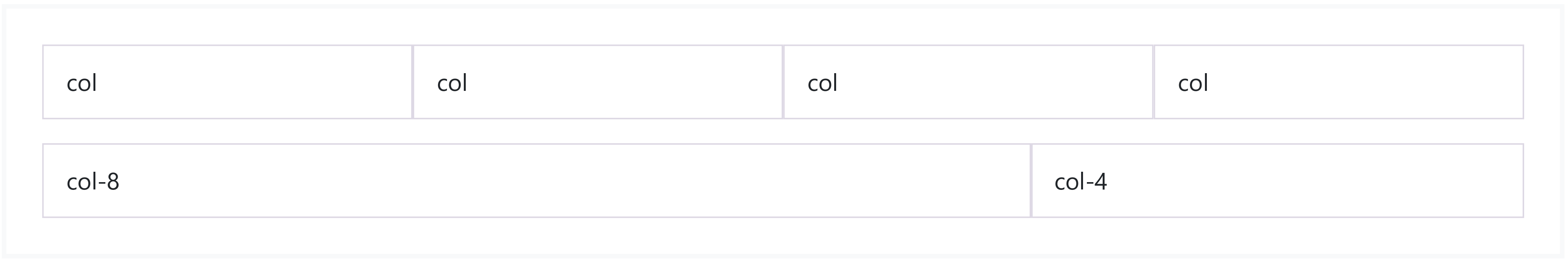
```
<div class="container">
  <div class="row justify-content-md-center">
    <div class="col col-lg-2">
      1 of 3
    </div>
    <div class="col-md-auto">
      Variable width content
    </div>
    <div class="col col-lg-2">
      3 of 3
    </div>
  </div>
  <div class="row">
    <div class="col">
      1 of 3
    </div>
    <div class="col-md-auto">
      Variable width content
    </div>
    <div class="col col-lg-2">
      3 of 3
    </div>
  </div>
</div>
```

Responsive classes

Bootstrap's grid includes five tiers of predefined classes for building complex responsive layouts. Customize the size of your columns on extra small, small, medium, large, or extra large devices however you see fit.

All breakpoints

For grids that are the same from the smallest of devices to the largest, use the `.col` and `.col-*` classes. Specify a numbered class when you need a particularly sized column; otherwise, feel free to stick to `.col`.

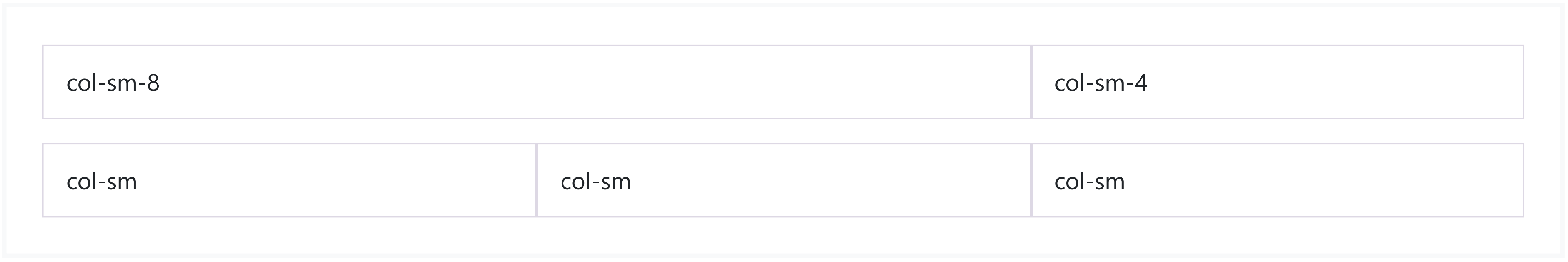


Copy

```
<div class="container">
  <div class="row">
    <div class="col">col</div>
    <div class="col">col</div>
    <div class="col">col</div>
    <div class="col">col</div>
  </div>
  <div class="row">
    <div class="col-8">col-8</div>
    <div class="col-4">col-4</div>
  </div>
</div>
```

Stacked to horizontal

Using a single set of `.col-sm-*` classes, you can create a basic grid system that starts out stacked and becomes horizontal at the small breakpoint (`sm`).



Copy

```
<div class="container">
  <div class="row">
    <div class="col-sm-8">col-sm-8</div>
    <div class="col-sm-4">col-sm-4</div>
  </div>
  <div class="row">
    <div class="col-sm">col-sm</div>
    <div class="col-sm">col-sm</div>
    <div class="col-sm">col-sm</div>
  </div>
</div>
```

Mix and match

Don't want your columns to simply stack in some grid tiers? Use a combination of different classes for each tier as needed. See the example below for a better idea of how it all works.

.col-md-8		.col-6 .col-md-4
.col-6 .col-md-4	.col-6 .col-md-4	.col-6 .col-md-4
.col-6	.col-6	

Copy

```
<div class="container">
  <!-- Stack the columns on mobile by making one full-width and the other half-width -->
  <div class="row">
    <div class="col-md-8">.col-md-8</div>
    <div class="col-6 col-md-4">.col-6 .col-md-4</div>
  </div>

  <!-- Columns start at 50% wide on mobile and bump up to 33.3% wide on desktop -->
  <div class="row">
    <div class="col-6 col-md-4">.col-6 .col-md-4</div>
    <div class="col-6 col-md-4">.col-6 .col-md-4</div>
    <div class="col-6 col-md-4">.col-6 .col-md-4</div>
  </div>

  <!-- Columns are always 50% wide, on mobile and desktop -->
  <div class="row">
    <div class="col-6">.col-6</div>
    <div class="col-6">.col-6</div>
  </div>
</div>
```

Gutters

Gutters can be responsively adjusted by breakpoint-specific padding and negative margin utility classes. To change the gutters in a given row, pair a negative margin utility on the `.row` and matching padding utilities on the `.col`s. The `.container` or `.container-fluid` parent may need to be adjusted too to avoid unwanted overflow, using again matching padding utility.

Here's an example of customizing the Bootstrap grid at the large (`lg`) breakpoint and above. We've increased the `.col` padding with `.px-lg-5`, counteracted that with `.mx-lg-n5` on the parent `.row` and then adjusted the `.container` wrapper with `.px-lg-5`.

Custom column padding	Custom column padding

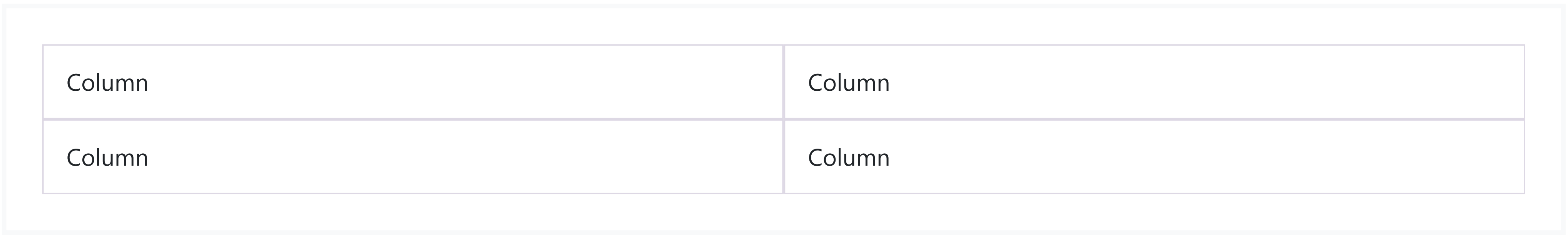
Copy

```html
<div class="container px-lg-5">
  <div class="row mx-lg-n5">
    <div class="col py-3 px-lg-5 border bg-light">Custom column padding</div>
    <div class="col py-3 px-lg-5 border bg-light">Custom column padding</div>
  </div>
</div>
```

Row columns

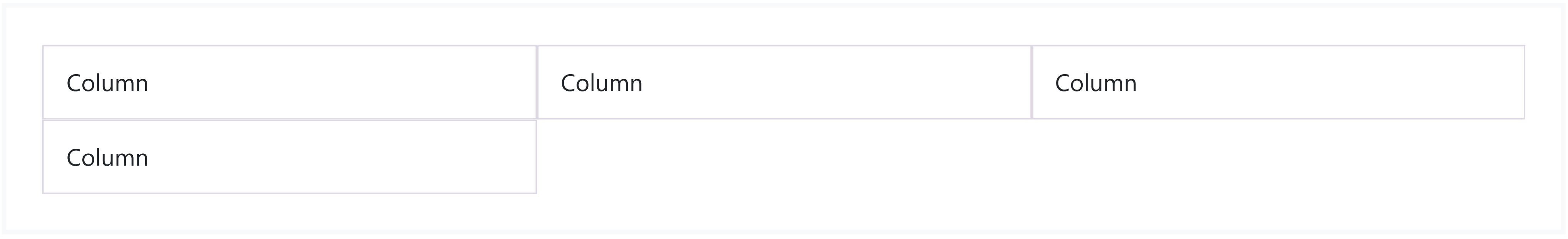
Use the responsive `.row-cols-*` classes to quickly set the number of columns that best render your content and layout. Whereas normal `.col-*` classes apply to the individual columns (e.g., `.col-md-4`), the row columns classes are set on the parent `.row` as a shortcut.

Use these row columns classes to quickly create basic grid layouts or to control your card layouts.



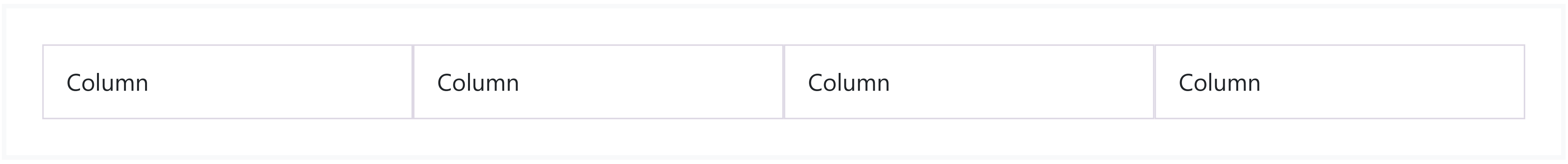
Copy

```html
<div class="container">
  <div class="row row-cols-2">
    <div class="col">Column</div>
    <div class="col">Column</div>
    <div class="col">Column</div>
    <div class="col">Column</div>
  </div>
</div>
```



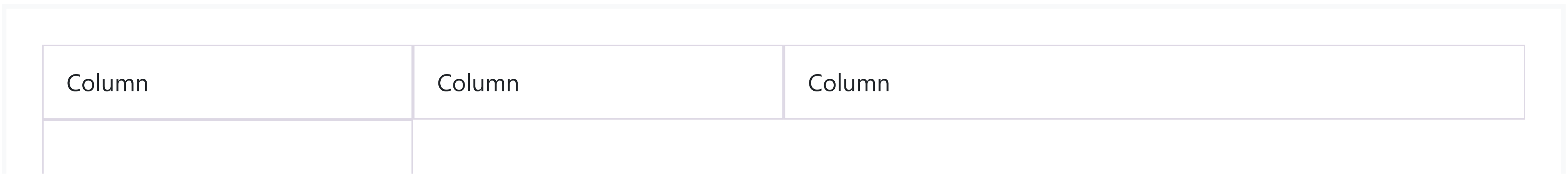
Copy

```html
<div class="container">
  <div class="row row-cols-3">
    <div class="col">Column</div>
    <div class="col">Column</div>
    <div class="col">Column</div>
    <div class="col">Column</div>
  </div>
</div>
```



Copy

```html
<div class="container">
  <div class="row row-cols-4">
    <div class="col">Column</div>
    <div class="col">Column</div>
    <div class="col">Column</div>
    <div class="col">Column</div>
  </div>
</div>
```



Column

Copy

```
<div class="container">
  <div class="row row-cols-4">
    <div class="col">Column</div>
    <div class="col">Column</div>
    <div class="col-6">Column</div>
    <div class="col">Column</div>
  </div>
</div>
```

Column	Column	Column	Column

Copy

```
<div class="container">
  <div class="row row-cols-1 row-cols-sm-2 row-cols-md-4">
    <div class="col">Column</div>
    <div class="col">Column</div>
    <div class="col">Column</div>
    <div class="col">Column</div>
  </div>
</div>
```

You can also use the accompanying Sass mixin, `row-cols()`:

Copy

```
.element {
  // Three columns to start
  @include row-cols(3);

  // Five columns from medium breakpoint up
  @include media-breakpoint-up(md) {
    @include row-cols(5);
  }
}
```

Alignment

Use flexbox alignment utilities to vertically and horizontally align columns. **Internet Explorer 10-11 do not support vertical alignment of flex items when the flex container has a `min-height` as shown below.** [See Flexbugs #3 for more details.](https://github.com/philipwalton/flexbugs#flexbug-3)

Vertical alignment

One of three columns	One of three columns	One of three columns

One of three columns	One of three columns	One of three columns

One of three columns | One of three columns | One of three columns

Copy

```
<div class="container">
  <div class="row align-items-start">
    <div class="col">
      One of three columns
    </div>
    <div class="col">
      One of three columns
    </div>
    <div class="col">
      One of three columns
    </div>
  </div>
  <div class="row align-items-center">
    <div class="col">
      One of three columns
    </div>
    <div class="col">
      One of three columns
    </div>
    <div class="col">
      One of three columns
    </div>
  </div>
  <div class="row align-items-end">
    <div class="col">
      One of three columns
    </div>
    <div class="col">
      One of three columns
    </div>
    <div class="col">
      One of three columns
    </div>
  </div>
</div>
```

One of three columns

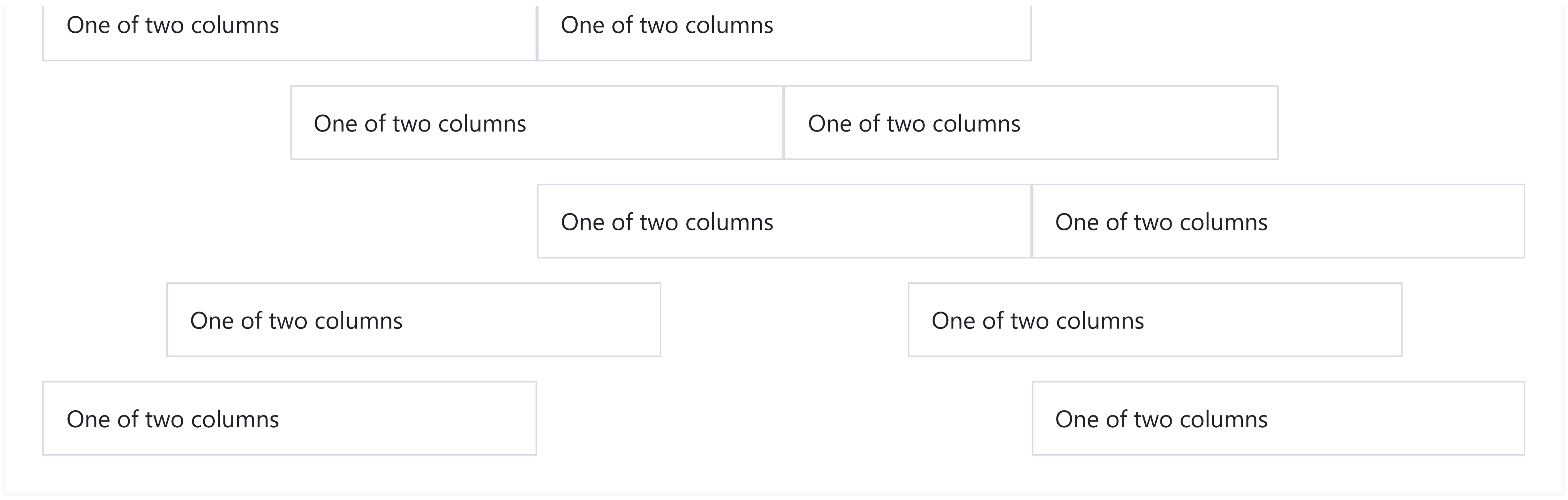
One of three columns

One of three columns

Copy

```
<div class="container">
  <div class="row">
    <div class="col align-self-start">
      One of three columns
    </div>
    <div class="col align-self-center">
      One of three columns
    </div>
    <div class="col align-self-end">
      One of three columns
    </div>
  </div>
</div>
```

Horizontal alignment



Copy

```
<div class="container">
  <div class="row justify-content-start">
    <div class="col-4">
      One of two columns
    </div>
    <div class="col-4">
      One of two columns
    </div>
  </div>
  <div class="row justify-content-center">
    <div class="col-4">
      One of two columns
    </div>
    <div class="col-4">
      One of two columns
    </div>
  </div>
  <div class="row justify-content-end">
    <div class="col-4">
      One of two columns
    </div>
    <div class="col-4">
      One of two columns
    </div>
  </div>
  <div class="row justify-content-around">
    <div class="col-4">
      One of two columns
    </div>
    <div class="col-4">
      One of two columns
    </div>
  </div>
  <div class="row justify-content-between">
    <div class="col-4">
      One of two columns
    </div>
    <div class="col-4">
      One of two columns
    </div>
  </div>
</div>
```

No gutters

The gutters between columns in our predefined grid classes can be removed with `.no-gutters`. This removes the negative `margin`s from `.row` and the horizontal `padding` from all immediate children columns.

Here's the source code for creating these styles. Note that column overrides are scoped to only the first children columns and are targeted via attribute selector. While this generates a more specific selector, column padding can still be further customized with spacing utilities.

Need an edge-to-edge design? Drop the parent `.container` or `.container-fluid`.

Copy

```
.no-gutters {
  margin-right: 0;
  margin-left: 0;

  > .col,
  > [class*="col-"] {
    padding-right: 0;
    padding-left: 0;
  }
}
```

In practice, here's how it looks. Note you can continue to use this with all other predefined grid classes (including column widths, responsive tiers, reorders, and more).

.col-sm-6 .col-md-8 | .col-6 .col-md-4

Copy

```
<div class="row no-gutters">
  <div class="col-sm-6 col-md-8">.col-sm-6 .col-md-8</div>
  <div class="col-6 col-md-4">.col-6 .col-md-4</div>
</div>
```

Column wrapping

If more than 12 columns are placed within a single row, each group of extra columns will, as one unit, wrap onto a new line.

.col-9

.col-4
Since 9 + 4 = 13 > 12, this 4-column-wide div gets wrapped onto a new line as one contiguous unit.

.col-6
Subsequent columns continue along the new line.

Copy

```
<div class="container">
  <div class="row">
    <div class="col-9">.col-9</div>
    <div class="col-4">.col-4<br>Since 9 + 4 = 13 &gt; 12, this 4-column-wide div gets wrapped onto a new line as one contiguous
unit.</div>
    <div class="col-6">.col-6<br>Subsequent columns continue along the new line.</div>
  </div>
</div>
```

Column breaks

Breaking columns to a new line in flexbox requires a small hack: add an element with `width: 100%` wherever you want to wrap your columns to a new line. Normally this is accomplished with multiple `.row`s, but not every implementation method can account for this.

.col-6 .col-sm-3 | .col-6 .col-sm-3

.col-6 .col-sm-3 | .col-6 .col-sm-3

Copy

```
<div class="container">
  <div class="row">
    <div class="col-6 col-sm-3">.col-6 .col-sm-3</div>
    <div class="col-6 col-sm-3">.col-6 .col-sm-3</div>

    <!-- Force next columns to break to new line -->
    <div class="w-100"></div>

    <div class="col-6 col-sm-3">.col-6 .col-sm-3</div>
    <div class="col-6 col-sm-3">.col-6 .col-sm-3</div>
  </div>
</div>
```

You may also apply this break at specific breakpoints with our responsive display utilities.

.col-6 .col-sm-4	.col-6 .col-sm-4
.col-6 .col-sm-4	.col-6 .col-sm-4

Copy

```
<div class="container">
  <div class="row">
    <div class="col-6 col-sm-4">.col-6 .col-sm-4</div>
    <div class="col-6 col-sm-4">.col-6 .col-sm-4</div>

    <!-- Force next columns to break to new line at md breakpoint and up -->
    <div class="w-100 d-none d-md-block"></div>

    <div class="col-6 col-sm-4">.col-6 .col-sm-4</div>
    <div class="col-6 col-sm-4">.col-6 .col-sm-4</div>
  </div>
</div>
```

Reordering

Order classes

Use `.order-` classes for controlling the **visual order** of your content. These classes are responsive, so you can set the `order` by breakpoint (e.g., `.order-1.order-md-2`). Includes support for `1` through `12` across all five grid tiers.

First in DOM, no order applied	Third in DOM, with an order of 1	Second in DOM, with a larger order

Copy

```
<div class="container">
  <div class="row">
    <div class="col">
      First in DOM, no order applied
    </div>
    <div class="col order-12">
      Second in DOM, with a larger order
    </div>
    <div class="col order-1">
      Third in DOM, with an order of 1
    </div>
  </div>
</div>
```

There are also responsive `.order-first` and `.order-last` classes that change the `order` of an element by applying `order: -1` and `order: 13` (`order: $columns + 1`), respectively. These classes can also be intermixed with the numbered `.order-*` classes as needed.

Third in DOM, ordered first	Second in DOM, unordered	First in DOM, ordered last

Copy

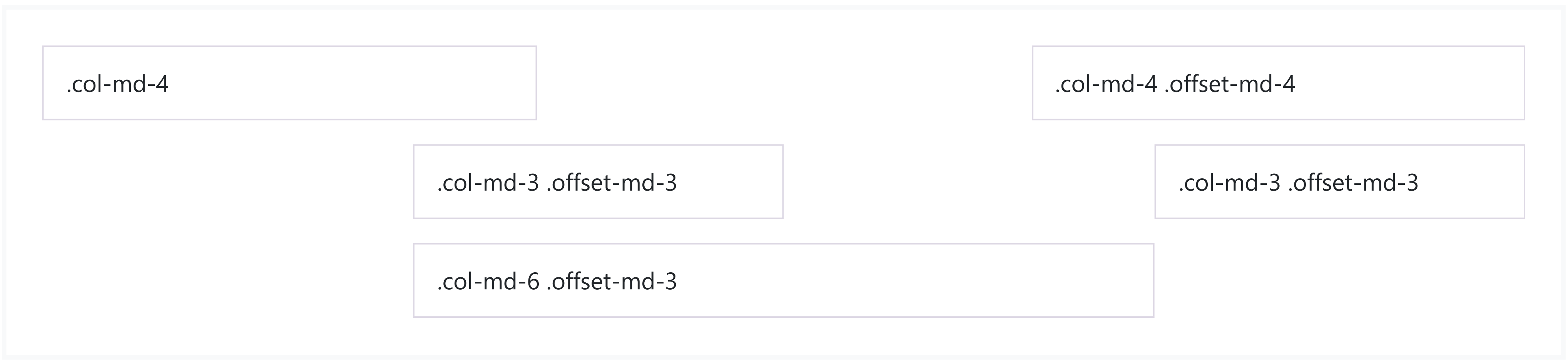
```
<div class="container">
  <div class="row">
    <div class="col order-last">
      First in DOM, ordered last
    </div>
    <div class="col">
      Second in DOM, unordered
    </div>
    <div class="col order-first">
      Third in DOM, ordered first
    </div>
  </div>
</div>
```

Offsetting columns

You can offset grid columns in two ways: our responsive `.offset-` grid classes and our [margin utilities](). Grid classes are sized to match columns while margins are more useful for quick layouts where the width of the offset is variable.

Offset classes

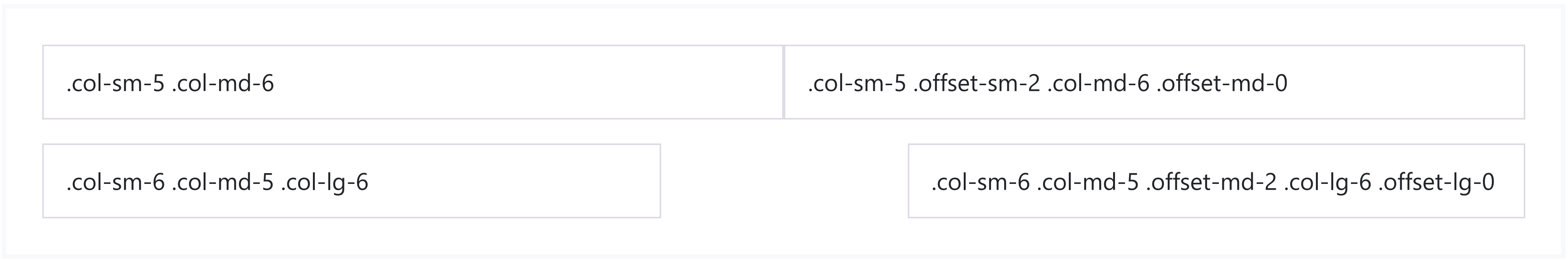
Move columns to the right using `.offset-md-*` classes. These classes increase the left margin of a column by `*` columns. For example, `.offset-md-4` moves `.col-md-4` over four columns.



Copy

```
<div class="container">
  <div class="row">
    <div class="col-md-4">.col-md-4</div>
    <div class="col-md-4 offset-md-4">.col-md-4 .offset-md-4</div>
  </div>
  <div class="row">
    <div class="col-md-3 offset-md-3">.col-md-3 .offset-md-3</div>
    <div class="col-md-3 offset-md-3">.col-md-3 .offset-md-3</div>
  </div>
  <div class="row">
    <div class="col-md-6 offset-md-3">.col-md-6 .offset-md-3</div>
  </div>
</div>
```

In addition to column clearing at responsive breakpoints, you may need to reset offsets. See this in action in [the grid example]().



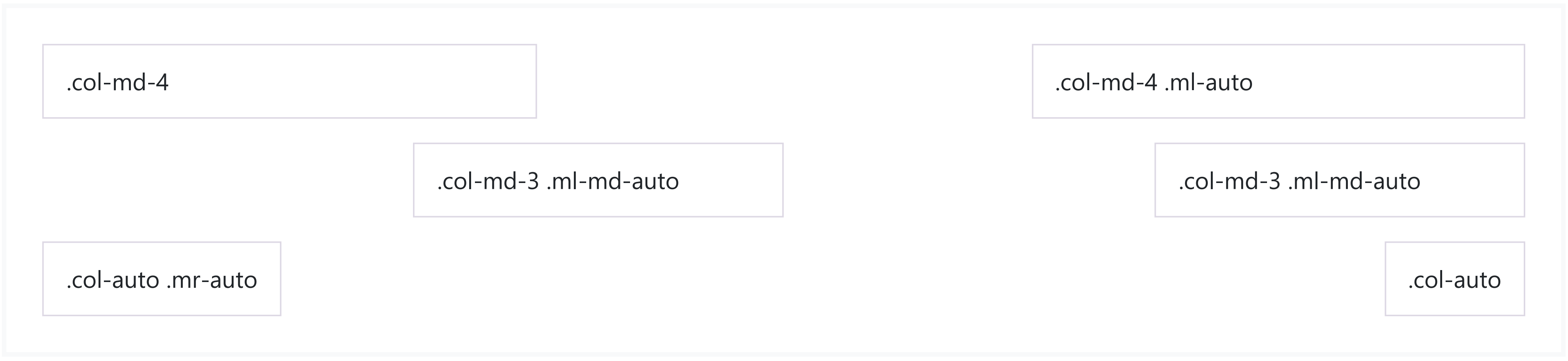
.col-sm-5 .col-md-6
.col-sm-5 .offset-sm-2 .col-md-6 .offset-md-0
.col-sm-6 .col-md-5 .col-lg-6
.col-sm-6 .col-md-5 .offset-md-2 .col-lg-6 .offset-lg-0

Copy

```
<div class="container">
  <div class="row">
    <div class="col-sm-5 col-md-6">.col-sm-5 .col-md-6</div>
    <div class="col-sm-5 offset-sm-2 col-md-6 offset-md-0">.col-sm-5 .offset-sm-2 .col-md-6 .offset-md-0</div>
  </div>
  <div class="row">
    <div class="col-sm-6 col-md-5 col-lg-6">.col-sm-6 .col-md-5 .col-lg-6</div>
    <div class="col-sm-6 col-md-5 offset-md-2 col-lg-6 offset-lg-0">.col-sm-6 .col-md-5 .offset-md-2 .col-lg-6 .offset-lg-0</div>
  </div>
</div>
```

Margin utilities

With the move to flexbox in v4, you can use margin utilities like `.mr-auto` to force sibling columns away from one another.

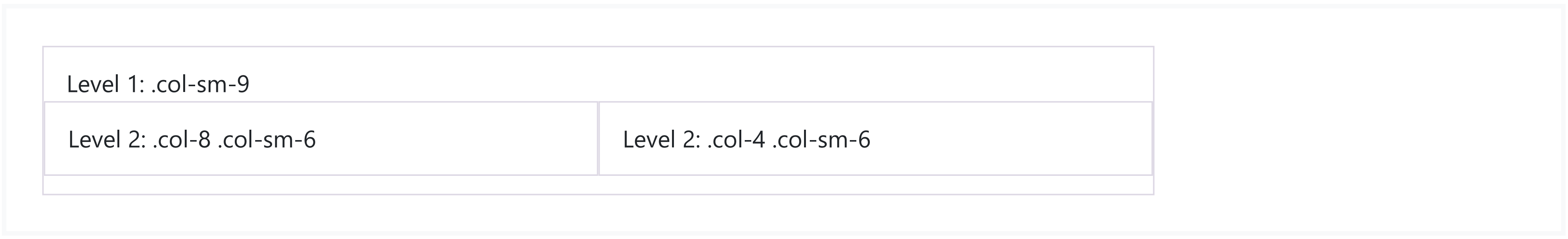

.col-md-4
.col-md-4 .ml-auto
.col-md-3 .ml-md-auto
.col-md-3 .ml-md-auto
.col-auto .mr-auto
.col-auto

Copy

```
<div class="container">
  <div class="row">
    <div class="col-md-4">.col-md-4</div>
    <div class="col-md-4 ml-auto">.col-md-4 .ml-auto</div>
  </div>
  <div class="row">
    <div class="col-md-3 ml-md-auto">.col-md-3 .ml-md-auto</div>
    <div class="col-md-3 ml-md-auto">.col-md-3 .ml-md-auto</div>
  </div>
  <div class="row">
    <div class="col-auto mr-auto">.col-auto .mr-auto</div>
    <div class="col-auto">.col-auto</div>
  </div>
</div>
```

Nesting

To nest your content with the default grid, add a new `.row` and set of `.col-sm-*` columns within an existing `.col-sm-*` column. Nested rows should include a set of columns that add up to 12 or fewer (it is not required that you use all 12 available columns).


Level 1: .col-sm-9
Level 2: .col-8 .col-sm-6
Level 2: .col-4 .col-sm-6

Copy

```
<div class="container">
  <div class="row">
    <div class="col-sm-9">
      Level 1: .col-sm-9
      <div class="row">
        <div class="col-8 col-sm-6">
          Level 2: .col-8 .col-sm-6
        </div>
        <div class="col-4 col-sm-6">
          Level 2: .col-4 .col-sm-6
        </div>
      </div>
    </div>
  </div>
</div>
```

Sass mixins

When using Bootstrap's source Sass files, you have the option of using Sass variables and mixins to create custom, semantic, and responsive page layouts. Our predefined grid classes use these same variables and mixins to provide a whole suite of ready-to-use classes for fast responsive layouts.

Variables

Variables and maps determine the number of columns, the gutter width, and the media query point at which to begin floating columns. We use these to generate the predefined grid classes documented above, as well as for the custom mixins listed below.

Copy

```
$grid-columns:      12;
$grid-gutter-width: 30px;

$grid-breakpoints: (
  // Extra small screen / phone
  xs: 0,
  // Small screen / phone
  sm: 576px,
  // Medium screen / tablet
  md: 768px,
  // Large screen / desktop
  lg: 992px,
  // Extra large screen / wide desktop
  xl: 1200px
);

$container-max-widths: (
  sm: 540px,
  md: 720px,
  lg: 960px,
  xl: 1140px
);
```

Mixins

Mixins are used in conjunction with the grid variables to generate semantic CSS for individual grid columns.

Copy

```
// Creates a wrapper for a series of columns
@include make-row();

// Make the element grid-ready (applying everything but the width)
@include make-col-ready();
@include make-col($size, $columns: $grid-columns);

// Get fancy by offsetting, or changing the sort order
@include make-col-offset($size, $columns: $grid-columns);
```

Example usage

You can modify the variables to your own custom values, or just use the mixins with their default values. Here's an example of using the default settings to create a two-column layout with a gap between.

Copy

```
.example-container {
  @include make-container();
  // Make sure to define this width after the mixin to override
  // `width: 100%` generated by `make-container()`
  width: 800px;
}

.example-row {
  @include make-row();
}

.example-content-main {
  @include make-col-ready();

  @include media-breakpoint-up(sm) {
    @include make-col(6);
  }
  @include media-breakpoint-up(lg) {
    @include make-col(8);
  }
}

.example-content-secondary {
  @include make-col-ready();

  @include media-breakpoint-up(sm) {
    @include make-col(6);
  }
  @include media-breakpoint-up(lg) {
    @include make-col(4);
  }
}
```

Main content Secondary content

Copy

```
<div class="example-container">
  <div class="example-row">
    <div class="example-content-main">Main content</div>
    <div class="example-content-secondary">Secondary content</div>
  </div>
</div>
```

Customizing the grid

Using our built-in grid Sass variables and maps, it's possible to completely customize the predefined grid classes. Change the number of tiers, the media query dimensions, and the container widths—then recompile.

Columns and gutters

The number of grid columns can be modified via Sass variables. `$grid-columns` is used to generate the widths (in percent) of each individual column while `$grid-gutter-width` sets the width for the column gutters.

Copy

```
$grid-columns: 12 !default;
$grid-gutter-width: 30px !default;
```

Grid tiers

Moving beyond the columns themselves, you may also customize the number of grid tiers. If you wanted just four grid tiers, you'd update the `$grid-breakpoints` and `$container-max-widths` to something like this:

Copy

```
$grid-breakpoints: (
  xs: 0,
  sm: 480px,
  md: 768px,
  lg: 1024px
);

$container-max-widths: (
  sm: 420px,
  md: 720px,
  lg: 960px
);
```

When making any changes to the Sass variables or maps, you’ll need to save your changes and recompile. Doing so will output a brand new set of predefined grid classes for column widths, offsets, and ordering. Responsive visibility utilities will also be updated to use the custom breakpoints. Make sure to set grid values in `px` (not `rem`, `em`, or `%`).

Spacing

Bootstrap includes a wide range of shorthand responsive margin and padding utility classes to modify an element's appearance.

How it works

Assign responsive-friendly `margin` or `padding` values to an element or a subset of its sides with shorthand classes. Includes support for individual properties, all properties, and vertical and horizontal properties. Classes are built from a default Sass map ranging from `.25rem` to `3rem`.

Notation

Spacing utilities that apply to all breakpoints, from `xs` to `xl`, have no breakpoint abbreviation in them. This is because those classes are applied from `min-width: 0` and up, and thus are not bound by a media query. The remaining breakpoints, however, do include a breakpoint abbreviation.

The classes are named using the format `{property}{sides}-{size}` for `xs` and `{property}{sides}-{breakpoint}-{size}` for `sm`, `md`, `lg`, and `xl`.

Where *property* is one of:

- `m` - for classes that set `margin`
- `p` - for classes that set `padding`

Where *sides* is one of:

- `t` - for classes that set `margin-top` or `padding-top`
- `b` - for classes that set `margin-bottom` or `padding-bottom`
- `l` - for classes that set `margin-left` or `padding-left`
- `r` - for classes that set `margin-right` or `padding-right`
- `x` - for classes that set both `*-left` and `*-right`
- `y` - for classes that set both `*-top` and `*-bottom`
- blank - for classes that set a `margin` or `padding` on all 4 sides of the element

Where *size* is one of:

- `0` - for classes that eliminate the `margin` or `padding` by setting it to `0`
- `1` - (by default) for classes that set the `margin` or `padding` to `$spacer * .25`
- `2` - (by default) for classes that set the `margin` or `padding` to `$spacer * .5`
- `3` - (by default) for classes that set the `margin` or `padding` to `$spacer`
- `4` - (by default) for classes that set the `margin` or `padding` to `$spacer * 1.5`
- `5` - (by default) for classes that set the `margin` or `padding` to `$spacer * 3`
- `auto` - for classes that set the `margin` to auto

(You can add more sizes by adding entries to the `$spacers` Sass map variable.)

Examples

Here are some representative examples of these classes:

Copy

```
.mt-0 {
  margin-top: 0 !important;
}

.ml-1 {
  margin-left: ($spacer * .25) !important;
}

.px-2 {
  padding-left: ($spacer * .5) !important;
  padding-right: ($spacer * .5) !important;
}

.p-3 {
  padding: $spacer !important;
}
```

Horizontal centering

Additionally, Bootstrap also includes an `.mx-auto` class for horizontally centering fixed-width block level content—that is, content that has `display: block` and a `width` set—by setting the horizontal margins to `auto`.

Centered element

Copy

```
<div class="mx-auto" style="width: 200px;">
  Centered element
</div>
```

Negative margin

In CSS, `margin` properties can utilize negative values (`padding` cannot). As of 4.2, we've added negative margin utilities for every non-zero integer size listed above (e.g., `1`, `2`, `3`, `4`, `5`). These utilities are ideal for customizing grid column gutters across breakpoints.

The syntax is nearly the same as the default, positive margin utilities, but with the addition of `n` before the requested size. Here's an example class that's the opposite of `.mt-1`:

Copy

```
.mt-n1 {
  margin-top: -0.25rem !important;
}
```

Here's an example of customizing the Bootstrap grid at the medium (`md`) breakpoint and above. We've increased the `.col` padding with `.px-md-5` and then counteracted that with `.mx-md-n5` on the parent `.row`.

Custom column padding

Custom column padding

Copy

```
<div class="row mx-md-n5">
  <div class="col px-md-5"><div class="p-3 border bg-light">Custom column padding</div></div>
  <div class="col px-md-5"><div class="p-3 border bg-light">Custom column padding</div></div>
</div>
```

Sizing

Easily make an element as wide or as tall with our width and height utilities.

Relative to the parent

Width and height utilities are generated from the `$sizes` Sass map in `_variables.scss`. Includes support for `25%`, `50%`, `75%`, `100%`, and `auto` by default. Modify those values as you need to generate different utilities here.

Width 25%

Width 50%

Width 75%

Width 100%

Width auto

Copy

```
<div class="w-25 p-3" style="background-color: #eee;">Width 25%</div>
<div class="w-50 p-3" style="background-color: #eee;">Width 50%</div>
<div class="w-75 p-3" style="background-color: #eee;">Width 75%</div>
<div class="w-100 p-3" style="background-color: #eee;">Width 100%</div>
<div class="w-auto p-3" style="background-color: #eee;">Width auto</div>
```

Height 25% Height 50% Height 75% Height 100% Height auto

Copy

```
<div style="height: 100px; background-color: rgba(255,0,0,0.1);">
  <div class="h-25 d-inline-block" style="width: 120px; background-color: rgba(0,0,255,.1)">Height 25%</div>
  <div class="h-50 d-inline-block" style="width: 120px; background-color: rgba(0,0,255,.1)">Height 50%</div>
  <div class="h-75 d-inline-block" style="width: 120px; background-color: rgba(0,0,255,.1)">Height 75%</div>
  <div class="h-100 d-inline-block" style="width: 120px; background-color: rgba(0,0,255,.1)">Height 100%</div>
  <div class="h-auto d-inline-block" style="width: 120px; background-color: rgba(0,0,255,.1)">Height auto</div>
</div>
```

You can also use `max-width: 100%;` and `max-height: 100%;` utilities as needed.

Copy

```
<img src="..." class="mw-100" alt="...">
```

Max-height
100%

Copy

```
<div style="height: 100px; background-color: rgba(255,0,0,0.1);">
  <div class="mh-100" style="width: 100px; height: 200px; background-color: rgba(0,0,255,0.1);">Max-height 100%</div>
</div>
```

Relative to the viewport

You can also use utilities to set the width and height relative to the viewport.

Copy

```
<div class="min-vw-100">Min-width 100vw</div>
<div class="min-vh-100">Min-height 100vh</div>
<div class="vw-100">Width 100vw</div>
<div class="vh-100">Height 100vh</div>
```

Colors

Convey meaning through color with a handful of color utility classes. Includes support for styling links with hover states, too.

Color

.text-primary

.text-secondary

.text-success

.text-danger

.text-warning

.text-info

.text-light

.text-dark

.text-body

.text-muted

.text-white

.text-black-50

.text-white-50

Copy

```
<p class="text-primary">.text-primary</p>
<p class="text-secondary">.text-secondary</p>
<p class="text-success">.text-success</p>
<p class="text-danger">.text-danger</p>
<p class="text-warning">.text-warning</p>
<p class="text-info">.text-info</p>
<p class="text-light bg-dark">.text-light</p>
<p class="text-dark">.text-dark</p>
<p class="text-body">.text-body</p>
<p class="text-muted">.text-muted</p>
<p class="text-white bg-dark">.text-white</p>
<p class="text-black-50">.text-black-50</p>
<p class="text-white-50 bg-dark">.text-white-50</p>
```

Contextual text classes also work well on anchors with the provided hover and focus states. **Note that the `.text-white` and `.text-muted` class has no additional link styling beyond underline.**

Primary link

Secondary link

Success link

Danger link

Warning link

Info link

Light link

Dark link

Muted link

White link

Copy

```
<p><a href="#" class="text-primary">Primary link</a></p>
<p><a href="#" class="text-secondary">Secondary link</a></p>
<p><a href="#" class="text-success">Success link</a></p>
<p><a href="#" class="text-danger">Danger link</a></p>
<p><a href="#" class="text-warning">Warning link</a></p>
<p><a href="#" class="text-info">Info link</a></p>
<p><a href="#" class="text-light bg-dark">Light link</a></p>
<p><a href="#" class="text-dark">Dark link</a></p>
<p><a href="#" class="text-muted">Muted link</a></p>
<p><a href="#" class="text-white bg-dark">White link</a></p>
```

Background color

Similar to the contextual text color classes, easily set the background of an element to any contextual class. Anchor components will darken on hover, just like the text classes. Background utilities **do not set** `color`, so in some cases you'll want to use `.text-*` utilities.

.bg-primary

.bg-secondary

.bg-success

.bg-danger

.bg-warning

.bg-info

.bg-light

.bg-dark

.bg-white

.bg-transparent

Copy

```
<div class="p-3 mb-2 bg-primary text-white">.bg-primary</div>
<div class="p-3 mb-2 bg-secondary text-white">.bg-secondary</div>
<div class="p-3 mb-2 bg-success text-white">.bg-success</div>
<div class="p-3 mb-2 bg-danger text-white">.bg-danger</div>
<div class="p-3 mb-2 bg-warning text-dark">.bg-warning</div>
<div class="p-3 mb-2 bg-info text-white">.bg-info</div>
<div class="p-3 mb-2 bg-light text-dark">.bg-light</div>
<div class="p-3 mb-2 bg-dark text-white">.bg-dark</div>
<div class="p-3 mb-2 bg-white text-dark">.bg-white</div>
<div class="p-3 mb-2 bg-transparent text-dark">.bg-transparent</div>
```

Background gradient

When `$enable-gradients` is set to `true` (default is `false`), you can use `.bg-gradient-` utility classes. Learn about our Sass options to enable these classes and more.

- `.bg-gradient-primary`
- `.bg-gradient-secondary`
- `.bg-gradient-success`
- `.bg-gradient-danger`
- `.bg-gradient-warning`
- `.bg-gradient-info`
- `.bg-gradient-light`
- `.bg-gradient-dark`

Dealing with specificity

Sometimes contextual classes cannot be applied due to the specificity of another selector. In some cases, a sufficient workaround is to wrap your element's content in a `<div>` with the class.

Conveying meaning to assistive technologies

Using color to add meaning only provides a visual indication, which will not be conveyed to users of assistive technologies – such as screen readers. Ensure that information denoted by the color is either obvious from the content itself (e.g. the visible text), or is included through alternative means, such as additional text hidden with the `.sr-only` class.

Text

Documentation and examples for common text utilities to control alignment, wrapping, weight, and more.

Text alignment

Easily realign text to components with text alignment classes.

Ambitioni dedisse scripsisse iudicaretur. Cras mattis iudicium purus sit amet fermentum. Donec sed odio operae, eu vulputate felis rhoncus. Praeterea iter est quasdam res quas ex communi. At nos hinc posthac, sitientis piros Afros. Petierunt uti sibi concilium totius Galliae in diem certam indicere. Cras mattis iudicium purus sit amet fermentum.

Copy

```
<p class="text-justify">Ambitioni dedisse scripsisse iudicaretur. Cras mattis iudicium purus sit amet fermentum. Donec sed odio operae, eu vulputate felis rhoncus. Praeterea iter
est quasdam res quas ex communi. At nos hinc posthac, sitientis piros Afros. Petierunt uti sibi concilium totius Galliae in diem certam indicere. Cras mattis iudicium purus sit
amet fermentum.</p>
```

For left, right, and center alignment, responsive classes are available that use the same viewport width breakpoints as the grid system.

Left aligned text on all viewport sizes.

Center aligned text on all viewport sizes.

Right aligned text on all viewport sizes.

Left aligned text on viewports sized SM (small) or wider.

Left aligned text on viewports sized MD (medium) or wider.

Left aligned text on viewports sized LG (large) or wider.

Left aligned text on viewports sized XL (extra-large) or wider.

Copy

```
<p class="text-left">Left aligned text on all viewport sizes.</p>
<p class="text-center">Center aligned text on all viewport sizes.</p>
<p class="text-right">Right aligned text on all viewport sizes.</p>

<p class="text-sm-left">Left aligned text on viewports sized SM (small) or wider.</p>
<p class="text-md-left">Left aligned text on viewports sized MD (medium) or wider.</p>
<p class="text-lg-left">Left aligned text on viewports sized LG (large) or wider.</p>
<p class="text-xl-left">Left aligned text on viewports sized XL (extra-large) or wider.</p>
```

Text wrapping and overflow

Wrap text with a `.text-wrap` class.

This text should wrap.

Copy

```
<div class="badge badge-primary text-wrap" style="width: 6rem;">
  This text should wrap.
</div>
```

Prevent text from wrapping with a `.text-nowrap` class.

This text should overflow the parent.

Copy

```
<div class="text-nowrap bd-highlight" style="width: 8rem;">
  This text should overflow the parent.
</div>
```

For longer content, you can add a `.text-truncate` class to truncate the text with an ellipsis. **Requires `display: inline-block` or `display: block`.**

Praeterea iter est quasdam ...
Praeterea iter est q...

Copy

```
<!-- Block level -->
<div class="row">
  <div class="col-2 text-truncate">
    Praeterea iter est quasdam res quas ex communi.
  </div>
</div>

<!-- Inline level -->
<span class="d-inline-block text-truncate" style="max-width: 150px;">
  Praeterea iter est quasdam res quas ex communi.
</span>
```

Word break

Prevent long strings of text from breaking your components' layout by using `.text-break` to set `word-wrap: break-word`.

mmmmmmmmmmmmmmmmmmmmmmmmmmmmmmmmmmmmmmmmmmmmmmmmmmmmmmmmmmmmmmmmmmmmmmmmmmmmmmmmmmmmmmmmmmmmmmmmmmmmmmm

Copy

```html
<p class="text-break">mmmmmmmmmmmmmmmmmmmmmmmmmmmmmmmmmmmmmmmmmmmmmmmmmmmmmmmmmmmmmmmmmmmmmmmmmmmmmmmmmmmmmmmmmmmmmmmmmmmmmmmm</p>
```

Text transform

Transform text in components with text capitalization classes.

lowercased text.

UPPERCASED TEXT.

CapiTaliZed Text.

Copy

```html
<p class="text-lowercase">Lowercased text.</p>
<p class="text-uppercase">Uppercased text.</p>
<p class="text-capitalize">CapiTaliZed text.</p>
```

Note how `.text-capitalize` only changes the first letter of each word, leaving the case of any other letters unaffected.

Font weight and italics

Quickly change the weight (boldness) of text or italicize text.

Bold text.

Bolder weight text (relative to the parent element).

Normal weight text.

Light weight text.

Lighter weight text (relative to the parent element).

Italic text.

Copy

```html
<p class="font-weight-bold">Bold text.</p>
<p class="font-weight-bolder">Bolder weight text (relative to the parent element).</p>
<p class="font-weight-normal">Normal weight text.</p>
<p class="font-weight-light">Light weight text.</p>
<p class="font-weight-lighter">Lighter weight text (relative to the parent element).</p>
<p class="font-italic">Italic text.</p>
```

Monospace

Change a selection to our monospace font stack with `.text-monospace`.

`This is in monospace`

Copy

```html
<p class="text-monospace">This is in monospace</p>
```

Reset color

Reset a text or link's color with `.text-reset`, so that it inherits the color from its parent.

Muted text with a reset link.

Copy

```html
<p class="text-muted">
  Muted text with a <a href="#" class="text-reset">reset link</a>.
</p>
```

Text decoration

Remove a text decoration with a `.text-decoration-none` class.

Non-underlined link

Copy

```html
<a href="#" class="text-decoration-none">Non-underlined link</a>
```