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

```scss
// 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

```scss
// 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

```scss
@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

```scss
// 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

```scss
@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

```scss
// 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

```scss
@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

```scss
$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.