Core concepts

- **Breakpoints are the building blocks of responsive design.** Use them to control when your layout can be adapted at a particular viewport or device size.
- **Use media queries to architect your CSS by breakpoint.** Media queries are a feature of CSS that allow you to conditionally apply styles based on a set of browser and operating system parameters. We most commonly use `min-width` in our media queries.
- **Mobile first, responsive design is the goal.** Bootstrap's CSS aims to apply the bare minimum of styles to make a layout work at the smallest breakpoint, and then layers on styles to adjust that design for larger devices. This optimizes your CSS, improves rendering time, and provides a great experience for your visitors.

Available breakpoints

Bootstrap includes six default breakpoints, sometimes referred to as *grid tiers*, for building responsively. These breakpoints can be customized if you're using our source Sass files.

Breakpoint	Class infix	Dimensions
X-Small	*None*	<576px
Small	`sm`	≥576px
Medium	`md`	≥768px
Large	`lg`	≥992px
Extra large	`xl`	≥1200px
Extra extra large	`xxl`	≥1400px

Each breakpoint was chosen to comfortably hold containers whose widths are multiples of 12. Breakpoints are also representative of a subset of common device sizes and viewport dimensions—they don't specifically target every use case or device. Instead, the ranges provide a strong and consistent foundation to build on for nearly any device.

These breakpoints are customizable via Sass—you'll find them in a Sass map in our `_variables.scss` stylesheet.

{{< scss-docs name="grid-breakpoints" file="scss/_variables.scss" >}}

For more information and examples on how to modify our Sass maps and variables, please refer to [the Sass section of the Grid documentation]({{< docsref "/layout/grid#sass" >}}).

Media queries

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.

Min-width

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

```
// Source mixins

// 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) { ... }
@include media-breakpoint-up(xxl) { ... }

// Usage

// 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;
  }
}
```

These Sass mixins translate in our compiled CSS using the values declared in our Sass variables. For example:

```
// X-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) { ... }

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

// XX-Large devices (larger desktops, 1400px and up)
@media (min-width: 1400px) { ... }
```

Max-width

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

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

```
@include media-breakpoint-down(lg) { ... }
@include media-breakpoint-down(xl) { ... }
@include media-breakpoint-down(xxl) { ... }

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

These mixins take those declared breakpoints, subtract `.02px` from them, and use them as our `max-width` values. For example:

```
// `xs` returns only a ruleset and no media query
// ... { ... }

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

// `md` applies to small devices (landscape phones, less than 768px)
@media (max-width: 767.98px) { ... }

// `lg` applies to medium devices (tablets, less than 992px)
@media (max-width: 991.98px) { ... }

// `xl` applies to large devices (desktops, less than 1200px)
@media (max-width: 1199.98px) { ... }

// `xxl` applies to x-large devices (large desktops, less than 1400px)
@media (max-width: 1399.98px) { ... }
```

{{< callout warning >}} {{< partial "callout-info-mediaqueries-breakpoints.md" >}} {{< /callout >}}

Single breakpoint

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

```
@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) { ... }
@include media-breakpoint-only(xxl) { ... }
```

For example the `@include media-breakpoint-only(md) { ... }` will result in :

```
@media (min-width: 768px) and (max-width: 991.98px) { ... }
```

Between breakpoints

Similarly, media queries may span multiple breakpoint widths:

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

Which results in:

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