

[View on GitHub](#)

# Buttons



Your new development career awaits.  
Check out the latest listings.

ads via Carbon

[On this page](#)

## Base class #

Bootstrap has a base `.btn` class that sets up basic styles such as padding and content alignment. By default, `.btn` controls have a transparent border and background color, and lack any explicit focus and hover styles.

### Base class

HTML



```
<button type="button" class="btn">Base class</button>
```

The `.btn` class is intended to be used in conjunction with our button variants, or to serve as a basis for your own custom styles.

If you are using the `.btn` class on its own, remember to at least define some explicit `:focus` and/or `:focus-visible` styles.

## Variants

Bootstrap includes several button variants, each serving its own semantic purpose, with a few extras thrown in for more control.



[Link](#)

HTML



```
<button type="button" class="btn btn-primary">Primary</button>
<button type="button" class="btn btn-secondary">Secondary</button>
<button type="button" class="btn btn-success">Success</button>
<button type="button" class="btn btn-danger">Danger</button>
<button type="button" class="btn btn-warning">Warning</button>
<button type="button" class="btn btn-info">Info</button>
<button type="button" class="btn btn-light">Light</button>
<button type="button" class="btn btn-dark">Dark</button>

<button type="button" class="btn btn-link">Link</button>
```

**Accessibility tip:** Using color to add meaning only provides a visual indication, which will not be conveyed to users of assistive technologies like screen readers. Please ensure the meaning is obvious from the content itself (e.g., the visible text with a [sufficient color contrast](#)) or is included through alternative means, such as additional text hidden with the `.visually-hidden` class.

## Disable text wrapping

If you don't want the button text to wrap, you can add the `.text-nowrap` class to the button. In Sass, you can set `$btn-white-space: nowrap` to disable text wrapping for each button.

## Button tags

The `.btn` classes are designed to be used with the `<button>` element. However, you can also use these classes on `<a>` or `<input>` elements (though some browsers may apply a slightly different rendering).

When using button classes on `<a>` elements that are used to trigger in-page functionality (like collapsing content), rather than linking to new pages or sections within the current page, these links should be given a `role="button"` to appropriately convey their purpose to assistive technologies such as screen readers.



---

HTML



```
<a class="btn btn-primary" href="#" role="button">Link</a>
<button class="btn btn-primary" type="submit">Button</button>
<input class="btn btn-primary" type="button" value="Input">
<input class="btn btn-primary" type="submit" value="Submit">
<input class="btn btn-primary" type="reset" value="Reset">
```

---

## Outline buttons

In need of a button, but not the hefty background colors they bring? Replace the default modifier classes with the `.btn-outline-*` ones to remove all background images and colors on any button.



---

HTML



```
<button type="button" class="btn btn-outline-primary">Primary</button>
<button type="button" class="btn btn-outline-secondary">Secondary</button>
<button type="button" class="btn btn-outline-success">Success</button>
<button type="button" class="btn btn-outline-danger">Danger</button>
<button type="button" class="btn btn-outline-warning">Warning</button>
<button type="button" class="btn btn-outline-info">Info</button>
<button type="button" class="btn btn-outline-light">Light</button>
<button type="button" class="btn btn-outline-dark">Dark</button>
```

Some of the button styles use a relatively light foreground color, and should only be used on a dark background in order to have sufficient contrast.

## Sizes

Fancy larger or smaller buttons? Add `.btn-lg` or `.btn-sm` for additional sizes.

Large button

Large button

HTML



```
<button type="button" class="btn btn-primary btn-lg">Large button</button>
<button type="button" class="btn btn-secondary btn-lg">Large button</button>
```

Small button


Small button

HTML



```
<button type="button" class="btn btn-primary btn-sm">Small button</button>
<button type="button" class="btn btn-secondary btn-sm">Small button</button>
```

You can even roll your own custom sizing with CSS variables:

A single button with the text "Custom button".

HTML



```
<button type="button" class="btn btn-primary"
      style="--bs-btn-padding-y: .25rem; --bs-btn-padding-x: .5rem; --bs-btn-font
Custom button
</button>
```

## Disabled state

Make buttons look inactive by adding the `disabled` boolean attribute to any `<button>` element. Disabled buttons have `pointer-events: none` applied to, preventing hover and active states from triggering.

A primary button with the text "Primary button".A secondary button with the text "Button".An outline primary button with the text "Primary button".An outline secondary button with the text "Button".

HTML



```
<button type="button" class="btn btn-primary" disabled>Primary button</button>
<button type="button" class="btn btn-secondary" disabled>Button</button>
<button type="button" class="btn btn-outline-primary" disabled>Primary button</butt
<button type="button" class="btn btn-outline-secondary" disabled>Button</button>
```

Disabled buttons using the `<a>` element behave a bit different:

- `<a>`s don't support the `disabled` attribute, so you must add the `.disabled` class to make it visually appear disabled.
- Some future-friendly styles are included to disable all `pointer-events` on anchor buttons.
- Disabled buttons using `<a>` should include the `aria-disabled="true"` attribute to indicate the state of the element to assistive technologies.
- Disabled buttons using `<a>` *should not* include the `href` attribute.

[Primary link](#)[Link](#)

HTML



```
<a class="btn btn-primary disabled" role="button" aria-disabled="true">Primary link</a>  
<a class="btn btn-secondary disabled" role="button" aria-disabled="true">Link</a>
```

## Link functionality caveat

To cover cases where you have to keep the `href` attribute on a disabled link, the `.disabled` class uses `pointer-events: none` to try to disable the link functionality of `<a>`s. Note that this CSS property is not yet standardized for HTML, but all modern browsers support it. In addition, even in browsers that do support `pointer-events: none`, keyboard navigation remains unaffected, meaning that sighted keyboard users and users of assistive technologies will still be able to activate these links. So to be safe, in addition to `aria-disabled="true"`, also include a `tabindex="-1"` attribute on these links to prevent them from receiving keyboard focus, and use custom JavaScript to disable their functionality altogether.

[Primary link](#)[Link](#)

HTML



```
<a href="#" class="btn btn-primary disabled" tabindex="-1" role="button" aria-disabled="true">Primary link</a>  
<a href="#" class="btn btn-secondary disabled" tabindex="-1" role="button" aria-disabled="true">Link</a>
```

## Block buttons

Create responsive stacks of full-width, “block buttons” like those in Bootstrap 4 with a mix of our display and gap utilities. By using utilities instead of button-specific classes, we have much greater control over spacing, alignment, and responsive behaviors.

Button

Button

HTML



```
<div class="d-grid gap-2">
  <button class="btn btn-primary" type="button">Button</button>
  <button class="btn btn-primary" type="button">Button</button>
</div>
```

Here we create a responsive variation, starting with vertically stacked buttons until the **md** breakpoint, where **.d-md-block** replaces the **.d-grid** class, thus nullifying the **gap-2** utility. Resize your browser to see them change.

Button

Button

HTML



```
<div class="d-grid gap-2 d-md-block">
  <button class="btn btn-primary" type="button">Button</button>
  <button class="btn btn-primary" type="button">Button</button>
</div>
```

You can adjust the width of your block buttons with grid column width classes. For example, for a half-width “block button”, use **.col-6**. Center it horizontally with **.mx-auto**, too.

Button

Button

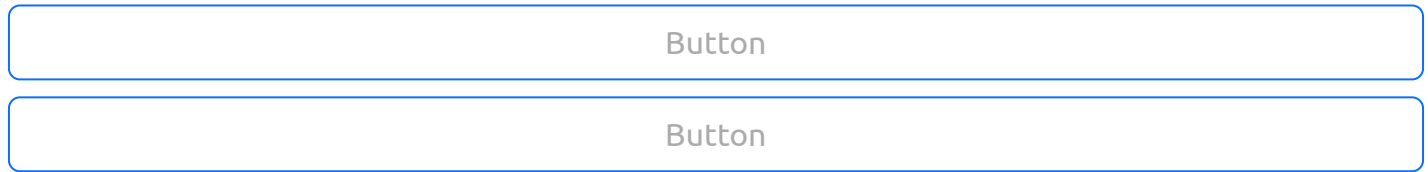
HTML



```
<div class="d-grid gap-2 col-6 mx-auto">
  <button class="btn btn-primary" type="button">Button</button>
</div>
```

```
<button class="btn btn-primary" type="button">Button</button>
</div>
```

Additional utilities can be used to adjust the alignment of buttons when horizontal. Here we've taken our previous responsive example and added some flex utilities and a margin utility on the button to right-align the buttons when they're no longer stacked.



HTML



```
<div class="d-grid gap-2 d-md-flex justify-content-md-end">
  <button class="btn btn-primary me-md-2" type="button">Button</button>
  <button class="btn btn-primary" type="button">Button</button>
</div>
```

## Button plugin

The button plugin allows you to create simple on/off toggle buttons.

Visually, these toggle buttons are identical to the [checkbox toggle buttons](#). However, they are conveyed differently by assistive technologies: the checkbox toggles will be announced by screen readers as “checked”/“not checked” (since, despite their appearance, they are fundamentally still checkboxes), whereas these toggle buttons will be announced as “button”/“button pressed”. The choice between these two approaches will depend on the type of toggle you are creating, and whether or not the toggle will make sense to users when announced as a checkbox or as an actual button.

## Toggle states

Add `data-bs-toggle="button"` to toggle a button's **active** state. If you're pre-toggling a button, you must manually add the `.active` class **and** `aria-pressed="true"` to ensure that it is conveyed appropriately to assistive technologies.



Toggle button

Active toggle button

Disabled toggle button

Toggle button

Active toggle button

Disabled toggle button

HTML



```
<p class="d-inline-flex gap-1">
  <button type="button" class="btn" data-bs-toggle="button">Toggle button</button>
  <button type="button" class="btn active" data-bs-toggle="button" aria-pressed="true">Active toggle button</button>
  <button type="button" class="btn disabled" data-bs-toggle="button">Disabled toggle button</button>
</p>
<p class="d-inline-flex gap-1">
  <button type="button" class="btn btn-primary" data-bs-toggle="button">Toggle button</button>
  <button type="button" class="btn btn-primary active" data-bs-toggle="button" aria-pressed="true">Active toggle button</button>
  <button type="button" class="btn btn-primary disabled" data-bs-toggle="button">Disabled toggle button</button>
</p>
```

Toggle link

Active toggle link

Disabled toggle link

Toggle link

Active toggle link

Disabled toggle link

HTML



```
<p class="d-inline-flex gap-1">
  <a href="#" class="btn" role="button" data-bs-toggle="button">Toggle link</a>
  <a href="#" class="btn active" role="button" data-bs-toggle="button" aria-pressed="true">Active toggle link</a>
  <a href="#" class="btn disabled" aria-disabled="true" role="button" data-bs-toggle="button">Disabled toggle link</a>
</p>
<p class="d-inline-flex gap-1">
  <a href="#" class="btn btn-primary" role="button" data-bs-toggle="button">Toggle link</a>
  <a href="#" class="btn btn-primary active" role="button" data-bs-toggle="button" aria-pressed="true">Active toggle link</a>
  <a href="#" class="btn btn-primary disabled" aria-disabled="true" role="button" data-bs-toggle="button">Disabled toggle link</a>
</p>
```

## Methods

You can create a button instance with the button constructor, for example:

```
const bsButton = new bootstrap.Button('#myButton')
```

Method	Description
<code>dispose</code>	Destroys an element's button. (Removes stored data on the DOM element)
<code>getInstance</code>	Static method which allows you to get the button instance associated with a DOM element, you can use it like this: <code>bootstrap.Button.getInstance(element).</code>
<code>getOrCreateInstance</code>	Static method which returns a button instance associated with a DOM element or creates a new one in case it wasn't initialized. You can use it like this: <code>bootstrap.Button.getOrCreateInstance(element).</code>
<code>toggle</code>	Toggles push state. Gives the button the appearance that it has been activated.

For example, to toggle all buttons

```
document.querySelectorAll('.btn').forEach(buttonElement => {  
  const button = bootstrap.Button.getOrCreateInstance(buttonElement)  
  button.toggle()  
})
```

## CSS

### Variables

Added in v5.2.0

As part of Bootstrap's evolving CSS variables approach, buttons now use local CSS variables on `.btn` for enhanced real-time customization. Values for the CSS variables are set via Sass, so Sass customization is still supported, too.

`scss/_buttons.scss`



```
--#{$prefix}btn-padding-x: #{$btn-padding-x};
--#{$prefix}btn-padding-y: #{$btn-padding-y};
--#{$prefix}btn-font-family: #{$btn-font-family};
@include rfs($btn-font-size, --#{$prefix}btn-font-size);
--#{$prefix}btn-font-weight: #{$btn-font-weight};
--#{$prefix}btn-line-height: #{$btn-line-height};
--#{$prefix}btn-color: #{$btn-color};
--#{$prefix}btn-bg: transparent;
--#{$prefix}btn-border-width: #{$btn-border-width};
--#{$prefix}btn-border-color: transparent;
--#{$prefix}btn-border-radius: #{$btn-border-radius};
--#{$prefix}btn-hover-border-color: transparent;
--#{$prefix}btn-box-shadow: #{$btn-box-shadow};
--#{$prefix}btn-disabled-opacity: #{$btn-disabled-opacity};
--#{$prefix}btn-focus-box-shadow: 0 0 0 #{$btn-focus-width} rgba(var(--#{$prefix}bt
```

Each `.btn-*` modifier class updates the appropriate CSS variables to minimize additional CSS rules with our `button-variant()`, `button-outline-variant()`, and `button-size()` mixins.

Here's an example of building a custom `.btn-*` modifier class as we do for the buttons unique to our docs by reassigning Bootstrap's CSS variables with a mixture of our own CSS and Sass variables.

Custom button

site/assets/scss/\_buttons.scss



```
.btn-bd-primary {
  --bs-btn-font-weight: 600;
  --bs-btn-color: var(--bs-white);
  --bs-btn-bg: var(--bd-violet-bg);
  --bs-btn-border-color: var(--bd-violet-bg);
  --bs-btn-hover-color: var(--bs-white);
  --bs-btn-hover-bg: #{shade-color($bd-violet, 10%)};
  --bs-btn-hover-border-color: #{shade-color($bd-violet, 10%)};
  --bs-btn-focus-shadow-rgb: var(--bd-violet-rgb);
  --bs-btn-active-color: var(--bs-btn-hover-color);
  --bs-btn-active-bg: #{shade-color($bd-violet, 20%)};
}
```

```
--bs-btn-active-border-color: #{shade-color($bd-violet, 20%)};
}
```

## Sass variables

scss/\_variables.scss



```
$btn-color: var(--#{ $prefix }body-color);
$btn-padding-y: $input-btn-padding-y;
$btn-padding-x: $input-btn-padding-x;
$btn-font-family: $input-btn-font-family;
$btn-font-size: $input-btn-font-size;
$btn-line-height: $input-btn-line-height;
$btn-white-space: null; // Set to `nowrap` to prevent text wrapping

$btn-padding-y-sm: $input-btn-padding-y-sm;
$btn-padding-x-sm: $input-btn-padding-x-sm;
$btn-font-size-sm: $input-btn-font-size-sm;

$btn-padding-y-lg: $input-btn-padding-y-lg;
$btn-padding-x-lg: $input-btn-padding-x-lg;
$btn-font-size-lg: $input-btn-font-size-lg;

$btn-border-width: $input-btn-border-width;

$btn-font-weight: $font-weight-normal;
$btn-box-shadow: inset 0 1px 0 rgba($white, .15), 0 1px 1px rgba($black, .15);
$btn-focus-width: $input-btn-focus-width;
$btn-focus-box-shadow: $input-btn-focus-box-shadow;
$btn-disabled-opacity: .65;
$btn-active-box-shadow: inset 0 3px 5px rgba($black, .125);

$btn-link-color: var(--#{ $prefix }link-color);
$btn-link-hover-color: var(--#{ $prefix }link-hover-color);
$btn-link-disabled-color: $gray-600;
$btn-link-focus-shadow-rgb: to-rgb(mix(color-contrast($link-color), $link-color,

// Allows for customizing button radius independently from global border radius
$btn-border-radius: var(--#{ $prefix }border-radius);
$btn-border-radius-sm: var(--#{ $prefix }border-radius-sm);
$btn-border-radius-lg: var(--#{ $prefix }border-radius-lg);

$btn-transition: color .15s ease-in-out, background-color .15s ease-in
```

```

$btn-hover-bg-shade-amount: 15%;
$btn-hover-bg-tint-amount: 15%;
$btn-hover-border-shade-amount: 20%;
$btn-hover-border-tint-amount: 10%;
$btn-active-bg-shade-amount: 20%;
$btn-active-bg-tint-amount: 20%;
$btn-active-border-shade-amount: 25%;
$btn-active-border-tint-amount: 10%;

```

## Sass mixins

There are three mixins for buttons: button and button outline variant mixins (both based on `$theme-colors`), plus a button size mixin.

scss/mixins/\_buttons.scss



```

@mixin button-variant(
  $background,
  $border,
  $color: color-contrast($background),
  $hover-background: if($color == $color-contrast-light, shade-color($background, $
  $hover-border: if($color == $color-contrast-light, shade-color($border, $btn-hove
  $hover-color: color-contrast($hover-background),
  $active-background: if($color == $color-contrast-light, shade-color($background,
  $active-border: if($color == $color-contrast-light, shade-color($border, $btn-act
  $active-color: color-contrast($active-background),
  $disabled-background: $background,
  $disabled-border: $border,
  $disabled-color: color-contrast($disabled-background)
) {
  --#{$prefix}btn-color: #{$color};
  --#{$prefix}btn-bg: #{$background};
  --#{$prefix}btn-border-color: #{$border};
  --#{$prefix}btn-hover-color: #{$hover-color};
  --#{$prefix}btn-hover-bg: #{$hover-background};
  --#{$prefix}btn-hover-border-color: #{$hover-border};
  --#{$prefix}btn-focus-shadow-rgb: #{to-rgb(mix($color, $border, 15%))};
  --#{$prefix}btn-active-color: #{$active-color};
  --#{$prefix}btn-active-bg: #{$active-background};
  --#{$prefix}btn-active-border-color: #{$active-border};
  --#{$prefix}btn-active-shadow: #{$btn-active-box-shadow};

```

```
--#{$prefix}btn-disabled-color: #{$disabled-color};
--#{$prefix}btn-disabled-bg: #{$disabled-background};
--#{$prefix}btn-disabled-border-color: #{$disabled-border};
}
```

---

scss/mixins/\_buttons.scss

---



```
@mixin button-outline-variant(
  $color,
  $color-hover: color-contrast($color),
  $active-background: $color,
  $active-border: $color,
  $active-color: color-contrast($active-background)
) {
  --#{$prefix}btn-color: #{$color};
  --#{$prefix}btn-border-color: #{$color};
  --#{$prefix}btn-hover-color: #{$color-hover};
  --#{$prefix}btn-hover-bg: #{$active-background};
  --#{$prefix}btn-hover-border-color: #{$active-border};
  --#{$prefix}btn-focus-shadow-rgb: #{to-rgb($color)};
  --#{$prefix}btn-active-color: #{$active-color};
  --#{$prefix}btn-active-bg: #{$active-background};
  --#{$prefix}btn-active-border-color: #{$active-border};
  --#{$prefix}btn-active-shadow: #{$btn-active-box-shadow};
  --#{$prefix}btn-disabled-color: #{$color};
  --#{$prefix}btn-disabled-bg: transparent;
  --#{$prefix}btn-disabled-border-color: #{$color};
  --#{$prefix}gradient: none;
}
```

---

scss/mixins/\_buttons.scss

---



```
@mixin button-size($padding-y, $padding-x, $font-size, $border-radius) {
  --#{$prefix}btn-padding-y: #{$padding-y};
  --#{$prefix}btn-padding-x: #{$padding-x};
  @include rfs($font-size, --#{$prefix}btn-font-size);
  --#{$prefix}btn-border-radius: #{$border-radius};
}
```

# Sass loops

Button variants (for regular and outline buttons) use their respective mixins with our `$theme-colors` map to generate the modifier classes in `scss/_buttons.scss`.

---

scss/\_buttons.scss



```
@each $color, $value in $theme-colors {  
  .btn-#{ $color } {  
    @if $color == "light" {  
      @include button-variant(  
        $value,  
        $value,  
        $hover-background: shade-color($value, $btn-hover-bg-shade-amount),  
        $hover-border: shade-color($value, $btn-hover-border-shade-amount),  
        $active-background: shade-color($value, $btn-active-bg-shade-amount),  
        $active-border: shade-color($value, $btn-active-border-shade-amount)  
      );  
    } @else if $color == "dark" {  
      @include button-variant(  
        $value,  
        $value,  
        $hover-background: tint-color($value, $btn-hover-bg-tint-amount),  
        $hover-border: tint-color($value, $btn-hover-border-tint-amount),  
        $active-background: tint-color($value, $btn-active-bg-tint-amount),  
        $active-border: tint-color($value, $btn-active-border-tint-amount)  
      );  
    } @else {  
      @include button-variant($value, $value);  
    }  
  }  
}  
  
@each $color, $value in $theme-colors {  
  .btn-outline-#{ $color } {  
    @include button-outline-variant($value);  
  }  
}
```

---

## Bootstrap

Designed and built with all the love in the world by the Bootstrap team with the help of our contributors.

Code licensed MIT, docs CC BY 3.0.

Currently v5.3.3.

### Links

[Home](#)

[Docs](#)

[Examples](#)

[Icons](#)

[Themes](#)

[Blog](#)

[Swag Store](#)

### Projects

[Bootstrap 5](#)

[Bootstrap 4](#)

[Icons](#)

[RFS](#)

[Examples repo](#)

### Guides

[Getting started](#)

[Starter template](#)

[Webpack](#)

[Parcel](#)

[Vite](#)

### Community

[Issues](#)

[Discussions](#)

[Corporate sponsors](#)

[Open Collective](#)

[Stack Overflow](#)