

## How it works

Here's what you need to know before getting started with the navbar:

- Navbars require a wrapping `.navbar` with `.navbar-expand{<sm|-md|-lg|-xl|-xxl>}` for responsive collapsing and [color scheme](#) classes.
- Navbars and their contents are fluid by default. Change the [container](#) to limit their horizontal width in different ways.
- Use our `[spacing]({{< docsref "/utilities/spacing" >}})` and `[flex]({{< docsref "/utilities/flex" >}})` utility classes for controlling spacing and alignment within navbars.
- Navbars are responsive by default, but you can easily modify them to change that. Responsive behavior depends on our Collapse JavaScript plugin.
- Ensure accessibility by using a `<nav>` element or, if using a more generic element such as a `<div>`, add a `role="navigation"` to every navbar to explicitly identify it as a landmark region for users of assistive technologies.
- Indicate the current item by using `aria-current="page"` for the current page or `aria-current="true"` for the current item in a set.

{{< callout info >}} {{< partial "callout-info-prefersreducedmotion.md" >}} {{< /callout >}}

## Supported content

Navbars come with built-in support for a handful of sub-components. Choose from the following as needed:

- `.navbar-brand` for your company, product, or project name.
- `.navbar-nav` for a full-height and lightweight navigation (including support for dropdowns).
- `.navbar-toggler` for use with our collapse plugin and other [navigation toggling](#) behaviors.
- Flex and spacing utilities for any form controls and actions.
- `.navbar-text` for adding vertically centered strings of text.
- `.collapse.navbar-collapse` for grouping and hiding navbar contents by a parent breakpoint.
- Add an optional `.navbar-scroll` to set a `max-height` and [scroll expanded navbar content](#).

Here's an example of all the sub-components included in a responsive light-themed navbar that automatically collapses at the `lg` (large) breakpoint.

{{< example >}}

Navbar ☐

- [Home](#)
- [Link](#)
- [Dropdown](#)
  - [Action](#)
  - [Another action](#)
  - [Something else here](#)
- Disabled

{{< /example >}}

This example uses `[background]({{< docsref "/utilities/background" >}})` (`bg-light`) and `[spacing]({{< docsref "/utilities/spacing" >}})` (`my-2`, `my-lg-0`, `me-sm-0`, `my-sm-0`) utility classes.

## Brand

The `.navbar-brand` can be applied to most elements, but an anchor works best, as some elements might require utility classes or custom styles.

### Text

Add your text within an element with the `.navbar-brand` class.

```
{{< example >}}
```

[Navbar](#)

Navbar

```
{{< /example >}}
```

### Image

You can replace the text within the `.navbar-brand` with an `<img>`.

```
{{< example >}}
```



```
{{< /example >}}
```

### Image and text

You can also make use of some additional utilities to add an image and text at the same time. Note the addition of `.d-inline-block` and `.align-text-top` on the `<img>`.

```
{{< example >}}
```



[Bootstrap](#)

```
{{< /example >}}
```

## Nav

Navbar navigation links build on our `.nav` options with their own modifier class and require the use of [toggler classes](#) for proper responsive styling. **Navigation in navbars will also grow to occupy as much horizontal space as possible** to keep your navbar contents securely aligned.

Add the `.active` class on `.nav-link` to indicate the current page.

Please note that you should also add the `aria-current` attribute on the active `.nav-link`.

```
{{< example >}}
```

[Navbar](#) ☐

- [Home](#)
- [Features](#)
- [Pricing](#)
- Disabled

```
{{< /example >}}
```

And because we use classes for our navs, you can avoid the list-based approach entirely if you like.

```
{{< example >}}
```

## Navbar

[Home](#) [Features](#) [Pricing](#) Disabled

```
{{< /example >}}
```

You can also use dropdowns in your navbar. Dropdown menus require a wrapping element for positioning, so be sure to use separate and nested elements for `.nav-item` and `.nav-link` as shown below.

```
{{< example >}}
```

## Navbar

- [Home](#)
- [Features](#)
- [Pricing](#)
- [Dropdown link](#)
  - [Action](#)
  - [Another action](#)
  - [Something else here](#)

```
{{< /example >}}
```

## Forms

Place various form controls and components within a navbar:

```
{{< example >}}
```

Search

```
{{< /example >}}
```

Immediate child elements of `.navbar` use flex layout and will default to `justify-content: space-between`. Use additional [flex utilities]({{< docsref "/utilities/flex" >}}) as needed to adjust this behavior.

```
{{< example >}}
```

Navbar

Search

```
{{< /example >}}
```

Input groups work, too. If your navbar is an entire form, or mostly a form, you can use the `<form>` element as the container and save some HTML.

```
{{< example >}}
```

@

```
{{< /example >}}
```

Various buttons are supported as part of these navbar forms, too. This is also a great reminder that vertical alignment utilities can be used to align different sized elements.

```
{{< example >}}
```

Main button

Smaller button

```
{{< /example >}}
```

## Text

Navbars may contain bits of text with the help of `.navbar-text`. This class adjusts vertical alignment and horizontal spacing for strings of text.

{{< example >}}

Navbar text with an inline element

{{< /example >}}

Mix and match with other components and utilities as needed.

{{< example >}}

[Navbar w/ text](#) 

- [Home](#)
- [Features](#)
- [Pricing](#)

Navbar text with an inline element

{{< /example >}}

## Color schemes


Theming the navbar has never been easier thanks to the combination of theming classes and `background-color` utilities. Choose from `.navbar-light` for use with light background colors, or `.navbar-dark` for dark background colors. Then, customize with `.bg-*` utilities.

[Navbar](#) 

- [Home](#)
- [Features](#)
- [Pricing](#)
- [About](#)

[Navbar](#) 

- [Home](#)
- [Features](#)
- [Pricing](#)
- [About](#)

[Navbar](#) 

- [Home](#)
- [Features](#)
- [Pricing](#)
- [About](#)

```
<nav class="navbar navbar-dark bg-dark">
  <!-- Navbar content -->
</nav>

<nav class="navbar navbar-dark bg-primary">
  <!-- Navbar content -->
</nav>

<nav class="navbar navbar-light" style="background-color: #e3f2fd;">
```

```
<!-- Navbar content -->
</nav>
```

## Containers

Although it's not required, you can wrap a navbar in a `.container` to center it on a page—though note that an inner container is still required. Or you can add a container inside the `.navbar` to only center the contents of a [fixed or static top navbar](#).

```
{{< example >}}
```

[Navbar](#)

```
{{< /example >}}
```

Use any of the responsive containers to change how wide the content in your navbar is presented.

```
{{< example >}}
```

[Navbar](#)

```
{{< /example >}}
```

## Placement

Use our [position utilities]({{< docsref "/utilities/position" >}}) to place navbars in non-static positions. Choose from fixed to the top, fixed to the bottom, or stickied to the top (scrolls with the page until it reaches the top, then stays there). Fixed navbars use `position: fixed`, meaning they're pulled from the normal flow of the DOM and may require custom CSS (e.g., `padding-top` on the `<body>`) to prevent overlap with other elements.

Also note that `.sticky-top` uses `position: sticky`, which [isn't fully supported in every browser](#).

```
{{< example >}}
```

[Default](#)

```
{{< /example >}}
```

```
{{< example >}}
```

[Fixed top](#)

```
{{< /example >}}
```

```
{{< example >}}
```

[Fixed bottom](#)

```
{{< /example >}}
```

```
{{< example >}}
```

[Sticky top](#)

```
{{< /example >}}
```


## Scrolling

Add `.navbar-nav-scroll` to a `.navbar-nav` (or other navbar sub-component) to enable vertical scrolling within the toggleable contents of a collapsed navbar. By default, scrolling kicks in at `75vh` (or 75% of the viewport height), but you can override that with the local CSS custom property `--bs-navbar-height` or custom styles. At larger viewports when the navbar is expanded, content will appear as it does in a default navbar.

Please note that this behavior comes with a potential drawback of `overflow` —when setting `overflow-y: auto` (required to scroll the content here), `overflow-x` is the equivalent of `auto`, which will crop some horizontal content.

Here's an example navbar using `.navbar-nav-scroll` with `style="--bs-scroll-height: 100px;"`, with some extra margin utilities for optimum spacing.

{{< example >}}

[Navbar scroll](#) 

- [Home](#)
- [Link](#)
- [Link](#)
  - [Action](#)
  - [Another action](#)
  - [Something else here](#)
- [Link](#)

{{< /example >}}

## Responsive behaviors

Navbars can use `.navbar-toggler`, `.navbar-collapse`, and `.navbar-expand{-sm|-md|-lg|-xl|-xxl}` classes to determine when their content collapses behind a button. In combination with other utilities, you can easily choose when to show or hide particular elements.

For navbars that never collapse, add the `.navbar-expand` class on the navbar. For navbars that always collapse, don't add any `.navbar-expand` class.

### Toggler

Navbar togglers are left-aligned by default, but should they follow a sibling element like a `.navbar-brand`, they'll automatically be aligned to the far right. Reversing your markup will reverse the placement of the toggler. Below are examples of different toggle styles.

With no `.navbar-brand` shown at the smallest breakpoint:

{{< example >}}



[Hidden brand](#)

- [Home](#)
- [Link](#)
- Disabled

{{< /example >}}

With a brand name shown on the left and toggler on the right:

{{< example >}}

[Navbar](#) 

- [Home](#)
- [Link](#)
- Disabled

{{< /example >}}

With a toggler on the left and brand name on the right:

{{< example >}}

- ☐
[Navbar](#)
- [Home](#)
  - [Link](#)
  - Disabled

{{< /example >}}

## External content

Sometimes you want to use the collapse plugin to trigger a container element for content that structurally sits outside of the `.navbar`. Because our plugin works on the `id` and `data-bs-target` matching, that's easily done!

{{< example >}}

### Collapsed content

Toggleable via the navbar brand.

☐

{{< /example >}}

When you do this, we recommend including additional JavaScript to move the focus programmatically to the container when it is opened. Otherwise, keyboard users and users of assistive technologies will likely have a hard time finding the newly revealed content - particularly if the container that was opened comes *before* the toggler in the document's structure. We also recommend making sure that the toggler has the `aria-controls` attribute, pointing to the `id` of the content container. In theory, this allows assistive technology users to jump directly from the toggler to the container it controls—but support for this is currently quite patchy.

## Offcanvas

Transform your expanding and collapsing navbar into an offcanvas drawer with the offcanvas plugin. We extend both the offcanvas default styles and use our `.navbar-expand-*` classes to create a dynamic and flexible navigation sidebar.

In the example below, to create an offcanvas navbar that is always collapsed across all breakpoints, omit the `.navbar-expand-*` class entirely.

{{< example >}}

[Offcanvas navbar](#) ☐

**Offcanvas**

- ☐
- [Home](#)
  - [Link](#)
  - [Dropdown](#)
    - [Action](#)

- [Another action](#)

- 

- [Something else here](#)

```
{{< /example >}}
```

To create an offcanvas navbar that expands into a normal navbar at a specific breakpoint like `lg`, use `.navbar-expand-lg`.

```
<nav class="navbar navbar-light navbar-expand-lg bg-light fixed-top">
  <a class="navbar-brand" href="#">Offcanvas navbar</a>
  <button class="navbar-toggler" type="button" data-bs-toggle="offcanvas" data-bs-target="#navbarOffcanvasLg" aria-controls="navbarOffcanvasLg">
    <span class="navbar-toggler-icon"></span>
  </button>
  <div class="offcanvas offcanvas-end" tabindex="-1" id="navbarOffcanvasLg" aria-labelledby="navbarOffcanvasLgLabel">
    ...
  </div>
</nav>
```

## Sass

### Variables

```
{{< scss-docs name="navbar-variables" file="scss/_variables.scss" >}}
```

```
{{< scss-docs name="navbar-theme-variables" file="scss/_variables.scss" >}}
```

### Loop

[Responsive navbar expand/collapse classes](#) (e.g., `.navbar-expand-lg`) are combined with the `$breakpoints` map and generated through a loop in `scss/_navbar.scss`.

```
{{< scss-docs name="navbar-expand-loop" file="scss/_navbar.scss" >}}
```