Docs Examples Icons Themes Blog

Search CTRL K

v5.3

CSS variables

Use Bootstrap's CSS custom properties for fast and forward-looking design and development.

View on GitHub

Bootstrap includes many CSS custom properties (variables) in its compiled CSS for real-time customization without the need to recompile Sass. These provide easy access to commonly used values like our theme colors, breakpoints, and primary font stacks when working in your browser's inspector, a code sandbox, or general prototyping.

All our custom properties are prefixed with `bs-` to avoid conflicts with third party CSS.

Root variables

Here are the variables we include (note that the `:root` is required) that can be accessed anywhere Bootstrap's CSS is loaded. They're located in our `_root.scss` file and included in our compiled dist files.

Default

These CSS variables are available everywhere, regardless of color mode.

```
:root,
[data-bs-theme=light] {
    --bs-blue: #0d6efd;
    --bs-indigo: #6610f2;
    --bs-purple: #6f42c1;
    --bs-pink: #d63384;
    --bs-red: #dc3545;
    --bs-orange: #fd7e14;
    --bs-yellow: #ffc107;
    --bs-green: #198754;
    --bs-teal: #20c997;
    --bs-cyan: #0dcaf0;
    --bs-black: #000;
    --bs-white: #fff;
    --bs-gray: #6c757d;
    --bs-gray-dark: #343a40;
    --bs-gray-100: #f8f9fa;
    --bs-gray-200: #e9ecef;
    --bs-gray-300: #dee2e6;
    --bs-gray-400: #ced4da;
    --bs-gray-500: #adb5bd;
    --bs-gray-600: #6c757d;
    --bs-gray-700: #495057;
    --bs-gray-800: #343a40;
    --bs-gray-900: #212529;
    --bs-primary: #0d6efd;
```

Component variables

Bootstrap 5 is increasingly making use of custom properties as local variables for various components. This way we reduce our compiled CSS, ensure styles aren't inherited in places like nested tables, and allow some basic restyling and extending of Bootstrap components after Sass compilation.

Have a look at our table documentation for some insight into how we're using CSS variables. Our navbars also use CSS variables as of v5.2.0. We're also using CSS variables across our grids—primarily for gutters with the new opt-in CSS grid—with more component usage coming in the future.

Whenever possible, we'll assign CSS variables at the base component level (e.g., `.navbar` for navbar and its sub-components). This reduces guessing on where and how to customize, and allows for easy modifications by our team in future updates.

Prefix

Most CSS variables use a prefix to avoid collisions with your own codebase. This prefix is in addition to the `--` that's required on every CSS variable.

Customize the prefix via the `$prefix` Sass variable. By default, it's set to `bs-` (note the trailing dash).

Examples

CSS variables offer similar flexibility to Sass's variables, but without the need for compilation before being served to the browser. For example, here we're resetting our page's font and link styles with CSS variables.

When I opened the downloaded Bootstrap folder, I found three main parts: **CSS**, **JS**, and optionally **Icons**. Inside the **CSS folder**, I saw files like bootstrap.css and bootstrap.min.css, which contain all the styling rules that control how my website looks — including layout, colors, spacing, buttons, forms, and responsiveness. The minified version is optimized for speed and commonly used in real websites. Moving to the **JS folder**, I noticed JavaScript files like bootstrap.bundle.min.js which include all the Bootstrap functionalities such as modals, dropdowns, tooltips, and more. This bundle file also includes Popper.js, which is essential for certain interactive components. Lastly, if I download Bootstrap Icons separately, the **Icons folder** contains a collection of scalable vector icons (SVGs) and font files, along with a bootstrap-icons.css file that lets me use icons easily in my code by applying classes like bi bi-alarm. Altogether, these folders provide everything needed to build a complete, responsive, and interactive web interface using Bootstrap.