Como personalizar Bootstrap con SASS

Diseño de Interfaces Web (DAW)

Julián Camilo Castro Restrepo

10-12-2024

ÍNDICE

1. 1.Descripcion general

Personalizar

Aprenda a tematizar, personalizar y ampliar Bootstrap con Sass, una gran cantidad de opciones globales, un sistema de colores expansivo y mucho más.

1.1. Descripción general

Existen varias formas de personalizar Bootstrap. La mejor opción puede depender de tu proyecto, la complejidad de tus herramientas de compilación, la versión de Bootstrap que estés usando, la compatibilidad del navegador y más.

Nuestros dos métodos preferidos son:

Usando Bootstrap a través del administrador de paquetes para que puedas usar y ampliar nuestros archivos fuente.

Usando los archivos de distribución compilados de Bootstrap o jsDelivr para que puedas agregar o anular los estilos de Bootstrap.

Si bien no podemos entrar en detalles aquí sobre cómo usar cada administrador de paquetes, podemos brindarle alguna orientación sobre el uso de Bootstrap con su propio compilador Sass .

Para quienes deseen utilizar los archivos de distribución, revisen la página de inicio para saber cómo incluir esos archivos y una página HTML de ejemplo. Desde allí, consulten la documentación para conocer el diseño, los componentes y los comportamientos que desean utilizar.

A medida que se familiarice con Bootstrap, continúe explorando esta sección para obtener más detalles sobre cómo utilizar nuestras opciones globales, cómo usar y cambiar nuestro sistema de colores, cómo construimos nuestros componentes, cómo usar nuestra creciente lista de propiedades CSS personalizadas y cómo optimizar su código al construir con Bootstrap.

1.2. CSP y SVG integrados

Varios componentes de Bootstrap incluyen SVG integrados en nuestro CSS para diseñar componentes de manera uniforme y sencilla en todos los navegadores y dispositivos. Para las organizaciones con configuraciones de CSP más estrictas, hemos documentado todas las instancias de nuestros SVG integrados (todos los cuales se aplican mediante background-image) para que pueda revisar sus opciones con más detalle.

Acordeón

Controles del carrusel

Botón de cerrar (usado en alertas y modales)

Casillas de verificación y botones de opción de formulario

Cambios de forma

Iconos de validación de formularios

Botones de alternancia de la barra de navegación

Seleccionar menús

Según las conversaciones de la comunidad, algunas opciones para abordar este problema en su propia base de código incluyen reemplazar las URL con recursos alojados localmente, eliminar las imágenes y usar imágenes en línea (no es posible en todos los componentes) y modificar su CSP. Nuestra recomendación es revisar cuidadosamente sus propias políticas de seguridad y decidir cuál es el mejor camino a seguir, si es necesario.

2. Estructura de archivo

Siempre que sea posible, evita modificar los archivos principales de Bootstrap. Para Sass, eso significa crear tu propia hoja de estilo que importe Bootstrap para que puedas modificarlo y extenderlo. Suponiendo que estés usando un administrador de paquetes como npm, tendrás una estructura de archivos que se verá así:

your-project/

├── scss/

│ └── custom.scss

└── node_modules/

│ └── bootstrap/

│ ├── js/

│ └── scss/

└── index.html

Si ha descargado nuestros archivos fuente y no está usando un administrador de paquetes, querrá crear manualmente algo similar a esa estructura, manteniendo los archivos fuente de Bootstrap separados de los suyos.

your-project/

├── scss/

│ └── custom.scss

├── bootstrap/

│ ├── js/

│ └── scss/

└── index.html

Importador

En tu custom.scss, importarás los archivos Sass de origen de Bootstrap. Tienes dos opciones: incluir todo Bootstrap o elegir las partes que necesitas. Te recomendamos esta última opción, aunque ten en cuenta que existen algunos requisitos y dependencias entre nuestros componentes. También tendrás que incluir algo de JavaScript para nuestros complementos.

```
// Custom.scss

// Option A: Include all of Bootstrap

// Include any default variable overrides here (though functions won't be available)


@import "../node_modules/bootstrap/scss/bootstrap";


// Then add additional custom code here
```

```
// Custom.scss
// Option B: Include parts of Bootstrap
// 1. Include functions first (so you can manipulate colors, SVGs, calc, etc)
@import "../node_modules/bootstrap/scss/functions";

// 2. Include any default variable overrides here
// 3. Include remainder of required Bootstrap stylesheets (including any separate color mode stylesheets)
@import "../node_modules/bootstrap/scss/variables";
@import "../node_modules/bootstrap/scss/variables-dark";
// 4. Include any default map overrides here

// 5. Include remainder of required parts
@import "../node_modules/bootstrap/scss/maps";
@import "../node_modules/bootstrap/scss/mixins";
@import "../node_modules/bootstrap/scss/root";

// 6. Optionally include any other parts as needed
@import "../node_modules/bootstrap/scss/utilities";
@import "../node_modules/bootstrap/scss/reboot";
@import "../node_modules/bootstrap/scss/type";
@import "../node_modules/bootstrap/scss/images";
@import "../node_modules/bootstrap/scss/containers";
@import "../node_modules/bootstrap/scss/grid";
@import "../node_modules/bootstrap/scss/helpers";
// 7. Optionally include utilities API last to generate classes based on the Sass map in `_utilities.scss`
@import "../node_modules/bootstrap/scss/utilities/api";

// 8. Add additional custom code here
```

Una vez que hayas configurado todo esto, puedes comenzar a modificar cualquiera de las variables y mapas de Sass en tu archivo custom.scss. También puedes comenzar a agregar partes de Bootstrap en la // Optionalsección según sea necesario. Te sugerimos que uses la pila de importación completa de nuestro bootstrap.scssarchivo como punto de partida.

2.1. Compilando

Para poder usar tu código Sass personalizado como CSS en el navegador, necesitas un compilador Sass. Sass se entrega como un paquete CLI, pero también puedes compilarlo con otras herramientas de compilación como Gulp o Webpack , o con aplicaciones GUI. Algunos IDE también tienen compiladores Sass integrados o como extensiones descargables.

Nos gusta usar la CLI para compilar nuestro Sass, pero puedes usar el método que prefieras. Desde la línea de comandos, ejecuta lo siguiente:

```
# Install Sass globally
npm install -g sass

# Watch your custom Sass for changes and compile it to CSS
sass --watch ./scss/custom.scss ./css/custom.css
```

Una vez compilado el CSS, puedes incluirlo en tus archivos HTML. Dentro de tu archivo, index.htmldeberás incluir el archivo CSS compilado. Asegúrate de actualizar la ruta del archivo CSS compilado si la has modificado.

```
<!doctype html>
<html lang="en">
  <head>
    <meta charset="utf-8">
    <meta name="viewport" content="width=device-width, initial-scale=1">
    <title>Custom Bootstrap</title>
    <link href="/css/custom.css" rel="stylesheet">
  </head>
  <body>
    <h1>Hello, world!</h1>
  </body>
</html>
```

Valores predeterminados de las variables

Cada variable Sass en Bootstrap incluye el !defaultindicador que te permite anular el valor predeterminado de la variable en tu propio Sass sin modificar el código fuente de Bootstrap. Copia y pega las variables según sea necesario, modifica sus valores y elimina el !defaultindicador. Si una variable ya ha sido asignada, no se volverá a asignar con los valores predeterminados en Bootstrap.

Encontrará la lista completa de variables de Bootstrap en scss/_variables.scss. Algunas variables están configuradas en null, estas variables no muestran la propiedad a menos que se anulen en su configuración.

Las anulaciones de variables deben realizarse después de que se importen nuestras funciones, pero antes del resto de las importaciones.

A continuación se muestra un ejemplo que cambia el background-colory colorpor el <body>al importar y compilar Bootstrap a través de npm:

```
// Required
@import "../node_modules/bootstrap/scss/functions";

// Default variable overrides
$body-bg: #000;
$body-color: #111;

// Required
@import "../node_modules/bootstrap/scss/variables";
@import "../node_modules/bootstrap/scss/variables-dark";
@import "../node_modules/bootstrap/scss/maps";
@import "../node_modules/bootstrap/scss/mixins";
@import "../node_modules/bootstrap/scss/root";

// Optional Bootstrap components here
@import "../node_modules/bootstrap/scss/reboot";
@import "../node_modules/bootstrap/scss/type";
// etc
```

Repita según sea necesario para cualquier variable en Bootstrap, incluidas las opciones globales a continuación.

¡Comienza a usar Bootstrap a través de npm con nuestro proyecto de inicio! Dirígete al repositorio de plantillas de ejemplo de Sass y JS para ver cómo crear y personalizar Bootstrap en tu propio proyecto npm. Incluye compilador Sass, Autoprefixer, Stylelint, PurgeCSS e íconos de Bootstrap.

Mapas y bucles

Bootstrap incluye un puñado de mapas Sass, pares clave-valor que facilitan la generación de familias de CSS relacionadas. Usamos mapas Sass para nuestros colores, puntos de interrupción de cuadrícula y más. Al igual que las variables Sass, todos los mapas Sass incluyen el !defaultindicador y se pueden anular y ampliar.

Algunos de nuestros mapas Sass se fusionan con mapas vacíos de forma predeterminada. Esto se hace para permitir una fácil expansión de un mapa Sass

determinado, pero tiene el costo de hacer que la eliminación de elementos de un mapa sea un poco más difícil.

Modificar mapa

Todas las variables del $theme-colorsmapa se definen como variables independientes. Para modificar un color existente en nuestro $theme-colorsmapa, agregue lo siguiente a su archivo Sass personalizado:

```
$primary: #0074d9;
$danger: #ff4136;
```

Posteriormente, estas variables se configuran en el $theme-colorsmapa de Bootstrap:

```
$theme-colors: (
  "primary": $primary,
  "danger": $danger
);
```

Añadir al mapa

Agrega nuevos colores a $theme-colors, o a cualquier otro mapa, creando un nuevo mapa Sass con tus valores personalizados y fusionándolo con el mapa original. En este caso, crearemos un nuevo $custom-colorsmapa y lo fusionaremos con $theme-colors.

```
// Create your own map
$custom-colors: (
  "custom-color": #900
);

// Merge the maps
$theme-colors: map-merge($theme-colors, $custom-colors);
```

Quitar del mapa

Para eliminar colores de $theme-colors, o cualquier otro mapa, utilice map-remove. Tenga en cuenta que debe insertar $theme-colorsentre nuestros requisitos justo después de su definición en variablesy antes de su uso en maps:

```
// Required
@import "../node_modules/bootstrap/scss/functions";
@import "../node_modules/bootstrap/scss/variables";
@import "../node_modules/bootstrap/scss/variables-dark";

$theme-colors: map-remove($theme-colors, "info", "light", "dark");

@import "../node_modules/bootstrap/scss/maps";
@import "../node_modules/bootstrap/scss/mixins";
@import "../node_modules/bootstrap/scss/root";

// Optional
@import "../node_modules/bootstrap/scss/reboot";
@import "../node_modules/bootstrap/scss/type";
// etc
```

Claves necesarias

Bootstrap asume la presencia de algunas claves específicas dentro de los mapas Sass, ya que las usamos y ampliamos nosotros mismos. A medida que personaliza los mapas incluidos, puede encontrar errores en los que se usa la clave de un mapa Sass específico.

Por ejemplo, usamos las claves primary, successy dangerde $theme-colorspara enlaces, botones y estados de formulario. Reemplazar los valores de estas claves no debería presentar problemas, pero eliminarlas puede causar problemas de compilación de Sass. En estos casos, deberá modificar el código de Sass que utiliza esos valores.

Además de los mapas Sass que tenemos, los colores del tema también se pueden usar como variables independientes, como $primary.

```
.custom-element {
  color: $gray-100;
  background-color: $dark;
}
```

Puedes aclarar u oscurecer los colores con las funciones tint-color()y de Bootstrap shade-color(). Estas funciones mezclarán colores con negro o blanco, a diferencia de las funciones lighten()y nativas de Sass darken()que cambiarán la claridad en una cantidad fija, lo que a menudo no produce el efecto deseado.

shift-color()combina estas dos funciones sombreando el color si el peso es positivo y tiñendo el color si el peso es negativo.

```
scss/_funciones.scss

// Tint a color: mix a color with white
@function tint-color($color, $weight) {
  @return mix(white, $color, $weight);
}

// Shade a color: mix a color with black
@function shade-color($color, $weight) {
  @return mix(black, $color, $weight);
}

// Shade the color if the weight is positive, else tint it
@function shift-color($color, $weight) {
  @return if($weight > 0, shade-color($color, $weight), tint-color($color, -$weight));
}
```

En la práctica, llamarías a la función y pasarías los parámetros de color y peso.

```
.custom-element {
  color: tint-color($primary, 10%);
}

.custom-element-2 {
  color: shade-color($danger, 30%);
}

.custom-element-3 {
  color: shift-color($success, 40%);
  background-color: shift-color($success, -60%);
}
```

Contraste de color

Para cumplir con los requisitos de contraste de las Pautas de Accesibilidad al Contenido Web (WCAG) , los autores deben proporcionar un contraste de color de texto mínimo de 4,5:1 y un contraste de color sin texto mínimo de 3:1 , con muy pocas excepciones.

Para ayudar con esto, incluimos la color-contrastfunción en Bootstrap. Utiliza el algoritmo de relación de contraste WCAG para calcular los umbrales de contraste en función de la luminancia relativa en un sRGBespacio de color para devolver automáticamente un color de contraste claro (#fff), oscuro (#212529) o negro (#000) en función del color base especificado. Esta función es especialmente útil para mixins o bucles en los que se generan varias clases.

Por ejemplo, para generar muestras de color a partir de nuestro $theme-colorsmapa:

```
@each $color, $value in $theme-colors {
  .swatch-#{$color} {
    color: color-contrast($value);
  }
}
También se puede utilizar para necesidades de contraste puntuales:

.custom-element {
  color: color-contrast(#000); // returns `color: #fff`
}
```

También puede especificar un color base con nuestras funciones de mapa de colores:

```
.custom-element {
  color: color-contrast($dark); // returns `color: #fff`
}
```

Escapar SVG

Usamos la escape-svgfunción para escapar los caracteres <, >y #para las imágenes de fondo SVG. Al usar la escape-svg función, las URL de datos deben estar entre comillas.

Funciones de suma y resta

Usamos las funciones addy subtract para encapsular la calc función CSS. El propósito principal de estas funciones es evitar errores cuando 0se pasa un valor “sin unidad” a una calc expresión. Expresiones como calc(10px - 0)devolverán un error en todos los navegadores, a pesar de ser matemáticamente correctas.

Ejemplo donde el cálculo es válido:

```
$border-radius: .25rem;
$border-width: 1px;

.element {
  // Output calc(.25rem - 1px) is valid
  border-radius: calc($border-radius - $border-width);
}

.element {
  // Output the same calc(.25rem - 1px) as above
  border-radius: subtract($border-radius, $border-width);
}
```

Ejemplo donde el cálculo no es válido:

```
$border-radius: .25rem;
$border-width: 0;

.element {
  // Output calc(.25rem - 0) is invalid
  border-radius: calc($border-radius - $border-width);
}

.element {
  // Output .25rem
  border-radius: subtract($border-radius, $border-width);
}
```

Mezclas

Nuestro scss/mixins/directorio tiene un montón de mixins que potencian partes de Bootstrap y también se pueden usar en su propio proyecto.

Esquemas de color

Hay disponible una combinación abreviada para la prefers-color-schemeconsulta de medios que admite los esquemas de color lighty dark. Consulte la documentación de modos de color para obtener información sobre nuestra combinación de modos de color.

```
scss/mixins/_esquema-de-color.scss

@mixin color-scheme($name) {
  @media (prefers-color-scheme: #{$name}) {
    @content;
  }
}

.custom-element {
  @include color-scheme(light) {
    // Insert light mode styles here
  }

  @include color-scheme(dark) {
    // Insert dark mode styles here
  }
}
```

Opciones

Personalice rápidamente Bootstrap con variables integradas para alternar fácilmente las preferencias CSS globales para controlar el estilo y el comportamiento.

Personalice Bootstrap con nuestro archivo de variables personalizado integrado y alterne fácilmente las preferencias de CSS globales con las nuevas `$enable-*`variables de Sass. Anule el valor de una variable y vuelva a compilar `npm run test`según sea necesario.

Puede encontrar y personalizar estas variables para opciones globales clave en `scss/_variables.scss`el archivo de Bootstrap

Variable	Valores	Descripción
$spacer	1rem(predeterminado) o cualquier valor > 0	Especifica el valor del espaciador predeterminado para generar programáticamente nuestras utilidades de espaciador .
$enable-dark-mode	true(predeterminado) ofalse	Habilita la compatibilidad con el modo oscuro integrado en todo el proyecto y sus componentes.
$enable-rounded	true(predeterminado) ofalse	Habilita border-radiusestilos predefinidos en varios componentes.
$enable-shadows	trueo false(predeterminado)	Habilita box-shadowestilos decorativos predefinidos en varios componentes. No afecta box-shadowlos símbolos utilizados para los estados de foco.
$enable-gradients	trueo false(predeterminado)	Habilita gradientes predefinidos a través de background-imageestilos en varios componentes.
$enable-transitions	true(predeterminado) ofalse	Habilita transitions predefinidos en varios componentes.
$enable-reduced-motion	true(predeterminado) ofalse	Habilita la prefers-reduced-motionconsulta de medios , que suprime ciertas animaciones/transiciones según las preferencias del navegador/sistema operativo de los usuarios.
$enable-grid-classes	true(predeterminado) ofalse	Permite la generación de clases CSS para el sistema de cuadrícula (por ejemplo .row, .col-md-1, , etc.).
$enable-container-classes	true(predeterminado) ofalse	Permite la generación de clases CSS para contenedores de diseño. (Novedad en la versión 5.2.0)
$enable-caret	true(predeterminado) ofalse	Habilita el cursor sobre pseudoelementos en .dropdown-toggle.
$enable-button-pointers	true(predeterminado) ofalse	Agregar un cursor en forma de “mano” a los elementos de botón no deshabilitados.
$enable-rfs	true(predeterminado) ofalse	Habilita RFS globalmente .
$enable-validation-icons	true(predeterminado) ofalse	Habilita background-imageíconos dentro de entradas de texto y algunos formularios personalizados para estados de validación.
$enable-negative-margins	trueo false(predeterminado)	Permite la generación de utilidades de margen negativo .
$enable-deprecation-messages	true(predeterminado) ofalse	Establezca esta opción falsepara ocultar las advertencias al usar cualquiera de las funciones y mixins obsoletos que se planea eliminar en v6.
$enable-important-utilities	true(predeterminado) ofalse	Habilita el !importantsufijo en clases de utilidad.
$enable-smooth-scroll	true(predeterminado) ofalse	Se aplica scroll-behavior: smoothglobalmente, excepto para los usuarios que solicitan movimiento reducido a través de prefers-reduced-motionuna consulta de medios.

Clases base

Los componentes de Bootstrap se construyen en gran medida con una nomenclatura de modificadores de base. Agrupamos tantas propiedades compartidas como sea posible en una clase base, como .btn, y luego agrupamos estilos individuales para cada variante en clases de modificadores, como .btn-primaryo .btn-success.

Para crear nuestras clases modificadoras, utilizamos @eachlos bucles de Sass para iterar sobre un mapa de Sass. Esto es especialmente útil para generar variantes de un componente $theme-colorsy crear variantes responsivas para cada punto de interrupción. A medida que personalizas estos mapas de Sass y vuelves a compilar, verás automáticamente tus cambios reflejados en estos bucles.

Consulta nuestra documentación de mapas y bucles de Sass para saber cómo personalizar estos bucles y ampliar el enfoque de modificadores de base de Bootstrap a tu propio código.

Modificadores

Muchos de los componentes de Bootstrap se construyen con un enfoque de clase modificadora base. Esto significa que la mayor parte del estilo está contenido en una clase base (por ejemplo, .btn) mientras que las variaciones de estilo se limitan a las clases modificadoras (por ejemplo, .btn-danger). Estas clases modificadoras se construyen a partir del $theme-colorsmapa para personalizar la cantidad y el nombre de nuestras clases modificadoras.

A continuación se muestran dos ejemplos de cómo recorremos el $theme-colorsmapa para generar modificadores para los componentes .alerty .list-group.

scss/_alerta.scss

```
// Generate contextual modifier classes for colorizing the alert
@each $state in map-keys($theme-colors) {
  .alert-#{$state} {
    --#{$prefix}alert-color: var(--#{$prefix}#{$state}-text-emphasis);
    --#{$prefix}alert-bg: var(--#{$prefix}#{$state}-bg-subtle);
    --#{$prefix}alert-border-color: var(--#{$prefix}#{$state}-border-subtle);
    --#{$prefix}alert-link-color: var(--#{$prefix}#{$state}-text-emphasis);
  }
}
```

scss/_lista-grupo.scss

```
// List group contextual variants
// Add modifier classes to change text and background color on individual items.
// Organizationally, this must come after the `:hover` states.

@each $state in map-keys($theme-colors) {
  .list-group-item-#{$state} {
    --#{$prefix}list-group-color: var(--#{$prefix}#{$state}-text-emphasis);
    --#{$prefix}list-group-bg: var(--#{$prefix}#{$state}-bg-subtle);
    --#{$prefix}list-group-border-color: var(--#{$prefix}#{$state}-border-subtle);
    --#{$prefix}list-group-action-hover-color: var(--#{$prefix}emphasis-color);
    --#{$prefix}list-group-action-hover-bg: var(--#{$prefix}#{$state}-border-subtle);
    --#{$prefix}list-group-action-active-color: var(--#{$prefix}emphasis-color);
    --#{$prefix}list-group-action-active-bg: var(--#{$prefix}#{$state}-border-subtle);
    --#{$prefix}list-group-active-color: var(--#{$prefix}#{$state}-bg-subtle);
    --#{$prefix}list-group-active-bg: var(--#{$prefix}#{$state}-text-emphasis);
    --#{$prefix}list-group-active-border-color:          var(--#{$prefix}#{$state}-text-
emphasis);
  }
}
```

Sensible

Estos bucles Sass no se limitan a los mapas de colores. También puedes generar variaciones interactivas de tus componentes. Por ejemplo, toma nuestra alineación interactiva de los menús desplegables, donde combinamos un @eachbucle para el $grid-breakpointsmapa Sass con una consulta de medios incluida.

```
scss/_dropdown.scss
// We deliberately hardcode the `bs-` prefix because we check
// this custom property in JS to determine Popper's positioning

@each $breakpoint in map-keys($grid-breakpoints) {
  @include media-breakpoint-up($breakpoint) {
    $infix: breakpoint-infix($breakpoint, $grid-breakpoints);

    .dropdown-menu#{$infix}-start {
      --bs-position: start;

      &[data-bs-popper] {
        right: auto;
        left: 0;
      }
    }
    .dropdown-menu#{$infix}-end {
      --bs-position: end;
      &[data-bs-popper] {
        right: 0;
        left: auto;
      }
    }
  }
}
```

Si modifica su $grid-breakpoints, sus cambios se aplicarán a todos los bucles que iteran sobre ese mapa.

```
scss/_variables.scss


$grid-breakpoints: (
  xs: 0,
  sm: 576px,
  md: 768px,
  lg: 992px,
  xl: 1200px,
  xxl: 1400px
);
```

Para obtener más información y ejemplos sobre cómo modificar nuestros mapas y variables Sass, consulte la sección CSS de la documentación de Grid .

Creando tu propio componente

Le recomendamos que adopte estas pautas al crear sus propios componentes con Bootstrap. Hemos ampliado este enfoque a los componentes personalizados en nuestra documentación y ejemplos. Los componentes como nuestros callouts se crean de la misma manera que los componentes que proporcionamos con clases base y modificadoras.

Este es un llamado a la acción. Lo creamos de manera personalizada para nuestros documentos, de modo que los mensajes que le enviamos se destaquen. Tiene tres variantes a través de clases modificadoras.

...

En tu CSS, tendrías algo como lo siguiente, donde la mayor parte del estilo se realiza mediante .callout. Luego, los estilos únicos entre cada variante se controlan mediante la clase modificadora.

```
// Base class
.callout {}


// Modifier classes
.callout-info {}
.callout-warning {}
.callout-danger {}
```

Variables raíz

Estas son las variables que incluimos (tenga en cuenta que son :rootobligatorias) a las que se puede acceder desde cualquier lugar donde se cargue el CSS de Bootstrap. Se encuentran en nuestro _root.scssarchivo y se incluyen en nuestros archivos de distribución compilados.

Por defecto

Estas variables CSS están disponibles en todas partes, independientemente del modo de color.

```
: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;
--bs-secondary: #6c757d;
--bs-success: #198754;
--bs-info: #0dcaf0;
--bs-warning: #ffc107;
--bs-danger: #dc3545;
--bs-light: #f8f9fa;
--bs-dark: #212529;
--bs-primary-rgb: 13, 110, 253;
--bs-secondary-rgb: 108, 117, 125;
--bs-success-rgb: 25, 135, 84;
--bs-info-rgb: 13, 202, 240;
--bs-warning-rgb: 255, 193, 7;
--bs-danger-rgb: 220, 53, 69;
--bs-light-rgb: 248, 249, 250;
--bs-dark-rgb: 33, 37, 41;
--bs-primary-text-emphasis: #052c65;
--bs-secondary-text-emphasis: #2b2f32;
--bs-success-text-emphasis: #0a3622;
--bs-info-text-emphasis: #055160;
--bs-warning-text-emphasis: #664d03;
--bs-danger-text-emphasis: #58151c;
--bs-light-text-emphasis: #495057;
--bs-dark-text-emphasis: #495057;
--bs-primary-bg-subtle: #cfe2ff;
--bs-secondary-bg-subtle: #e2e3e5;
--bs-success-bg-subtle: #d1e7dd;
--bs-info-bg-subtle: #cff4fc;
```

```
  --bs-warning-bg-subtle: #fff3cd;
  --bs-danger-bg-subtle: #f8d7da;
  --bs-light-bg-subtle: #fcfcfd;
  --bs-dark-bg-subtle: #ced4da;
  --bs-primary-border-subtle: #9ec5fe;
  --bs-secondary-border-subtle: #c4c8cb;
  --bs-success-border-subtle: #a3cfbb;
  --bs-info-border-subtle: #9eeaf9;
  --bs-warning-border-subtle: #ffe69c;
  --bs-danger-border-subtle: #f1aeb5;
  --bs-light-border-subtle: #e9ecef;
  --bs-dark-border-subtle: #adb5bd;
  --bs-white-rgb: 255, 255, 255;
  --bs-black-rgb: 0, 0, 0;
  --bs-font-sans-serif: system-ui, -apple-system, "Segoe UI", Roboto, "Helvetica Neue", "Noto Sans", "Liberation Sans", Arial, sans-serif, "Apple Color Emoji", "Segoe UI Emoji", "Segoe UI Symbol", "Noto Color Emoji";
  --bs-font-monospace: SFMono-Regular, Menlo, Monaco, Consolas, "Liberation Mono", "Courier New", monospace;
  --bs-gradient: linear-gradient(180deg, rgba(255, 255, 255, 0.15), rgba(255, 255, 255, 0));
  --bs-body-font-family: var(--bs-font-sans-serif);
  --bs-body-font-size: 1rem;
  --bs-body-font-weight: 400;
  --bs-body-line-height: 1.5;
  --bs-body-color: #212529;
  --bs-body-color-rgb: 33, 37, 41;
  --bs-body-bg: #fff;
  --bs-body-bg-rgb: 255, 255, 255;
  --bs-emphasis-color: #000;
  --bs-emphasis-color-rgb: 0, 0, 0;
  --bs-secondary-color: rgba(33, 37, 41, 0.75);
```

```
--bs-secondary-color-rgb: 33, 37, 41;
--bs-secondary-bg: #e9ecef;
--bs-secondary-bg-rgb: 233, 236, 239;
--bs-tertiary-color: rgba(33, 37, 41, 0.5);
--bs-tertiary-color-rgb: 33, 37, 41;
--bs-tertiary-bg: #f8f9fa;
--bs-tertiary-bg-rgb: 248, 249, 250;
--bs-heading-color: inherit;
--bs-link-color: #0d6efd;
--bs-link-color-rgb: 13, 110, 253;
--bs-link-decoration: underline;
--bs-link-hover-color: #0a58ca;
--bs-link-hover-color-rgb: 10, 88, 202;
--bs-code-color: #d63384;
--bs-highlight-color: #212529;
--bs-highlight-bg: #fff3cd;
--bs-border-width: 1px;
--bs-border-style: solid;
--bs-border-color: #dee2e6;
--bs-border-color-translucent: rgba(0, 0, 0, 0.175);
--bs-border-radius: 0.375rem;
--bs-border-radius-sm: 0.25rem;
--bs-border-radius-lg: 0.5rem;
--bs-border-radius-xl: 1rem;
--bs-border-radius-xxl: 2rem;
--bs-border-radius-2xl: var(--bs-border-radius-xxl);
--bs-border-radius-pill: 50rem;
--bs-box-shadow: 0 0.5rem 1rem rgba(0, 0, 0, 0.15);
--bs-box-shadow-sm: 0 0.125rem 0.25rem rgba(0, 0, 0, 0.075);
--bs-box-shadow-lg: 0 1rem 3rem rgba(0, 0, 0, 0.175);
```

```
  --bs-box-shadow-inset: inset 0 1px 2px rgba(0, 0, 0, 0.075);
  --bs-focus-ring-width: 0.25rem;
  --bs-focus-ring-opacity: 0.25;
  --bs-focus-ring-color: rgba(13, 110, 253, 0.25);
  --bs-form-valid-color: #198754;
  --bs-form-valid-border-color: #198754;
  --bs-form-invalid-color: #dc3545;
  --bs-form-invalid-border-color: #dc3545;
}
```

Modo oscuro

Estas variables están limitadas a nuestro modo oscuro incorporado.

```
[data-bs-theme=dark] {
  color-scheme: dark;
  --bs-body-color: #dee2e6;
  --bs-body-color-rgb: 222, 226, 230;
  --bs-body-bg: #212529;
  --bs-body-bg-rgb: 33, 37, 41;
  --bs-emphasis-color: #fff;
  --bs-emphasis-color-rgb: 255, 255, 255;
  --bs-secondary-color: rgba(222, 226, 230, 0.75);
  --bs-secondary-color-rgb: 222, 226, 230;
  --bs-secondary-bg: #343a40;
  --bs-secondary-bg-rgb: 52, 58, 64;
  --bs-tertiary-color: rgba(222, 226, 230, 0.5);
  --bs-tertiary-color-rgb: 222, 226, 230;
  --bs-tertiary-bg: #2b3035;
  --bs-tertiary-bg-rgb: 43, 48, 53;
  --bs-primary-text-emphasis: #6ea8fe;
```

```
--bs-secondary-text-emphasis: #a7acb1;
--bs-success-text-emphasis: #75b798;
--bs-info-text-emphasis: #6edff6;
--bs-warning-text-emphasis: #ffda6a;
--bs-danger-text-emphasis: #ea868f;
--bs-light-text-emphasis: #f8f9fa;
--bs-dark-text-emphasis: #dee2e6;
--bs-primary-bg-subtle: #031633;
--bs-secondary-bg-subtle: #161719;
--bs-success-bg-subtle: #051b11;
--bs-info-bg-subtle: #032830;
--bs-warning-bg-subtle: #332701;
--bs-danger-bg-subtle: #2c0b0e;
--bs-light-bg-subtle: #343a40;
--bs-dark-bg-subtle: #1a1d20;
--bs-primary-border-subtle: #084298;
--bs-secondary-border-subtle: #41464b;
--bs-success-border-subtle: #0f5132;
--bs-info-border-subtle: #087990;
--bs-warning-border-subtle: #997404;
--bs-danger-border-subtle: #842029;
--bs-light-border-subtle: #495057;
--bs-dark-border-subtle: #343a40;
--bs-heading-color: inherit;
--bs-link-color: #6ea8fe;
--bs-link-hover-color: #8bb9fe;
--bs-link-color-rgb: 110, 168, 254;
--bs-link-hover-color-rgb: 139, 185, 254;
--bs-code-color: #e685b5;
--bs-highlight-color: #dee2e6;
```

```
  --bs-highlight-bg: #664d03;
  --bs-border-color: #495057;
  --bs-border-color-translucent: rgba(255, 255, 255, 0.15);
  --bs-form-valid-color: #75b798;
  --bs-form-valid-border-color: #75b798;
  --bs-form-invalid-color: #ea868f;
  --bs-form-invalid-border-color: #ea868f;
}
```

Variables de componentes

Bootstrap 5 utiliza cada vez más propiedades personalizadas como variables locales para varios componentes. De esta manera, reducimos el CSS compilado, nos aseguramos de que los estilos no se hereden en lugares como tablas anidadas y permitimos algunos cambios de estilo y extensiones básicas de los componentes Bootstrap después de la compilación de Sass.

Eche un vistazo a nuestra documentación de tablas para obtener más información sobre cómo usamos las variables CSS . Nuestras barras de navegación también usan variables CSS a partir de la versión v5.2.0. También usamos variables CSS en nuestras cuadrículas (principalmente para los márgenes, la nueva cuadrícula CSS opcional) y en el futuro usaremos más componentes.

Siempre que sea posible, asignaremos variables CSS en el nivel del componente base (por ejemplo, .navbarpara la barra de navegación y sus subcomponentes). Esto reduce las dudas sobre dónde y cómo personalizar, y permite que nuestro equipo realice modificaciones fácilmente en futuras actualizaciones.

Prefijo

La mayoría de las variables CSS utilizan un prefijo para evitar colisiones con su propio código base. Este prefijo se suma al --que se requiere en todas las variables CSS.

Personaliza el prefijo mediante la $prefixvariable Sass. De forma predeterminada, está configurado como bs-(observa el guión final).

Ejemplos

Las variables CSS ofrecen una flexibilidad similar a las variables de Sass, pero sin necesidad de compilación antes de ser enviadas al navegador. Por ejemplo, aquí estamos restableciendo los estilos de fuente y de enlace de nuestra página con variables CSS.

```
body {
  font: 1rem/1.5 var(--bs-font-sans-serif);
}
a {
  color: var(--bs-blue);
}
```

Variables de enfoque

Agregado en v5.3.0

Bootstrap proporciona estilos personalizados :focusmediante una combinación de variables Sass y CSS que se pueden agregar opcionalmente a componentes y elementos específicos. Aún no anulamos todos :focuslos estilos de forma global.

En nuestro Sass, establecemos valores predeterminados que se pueden personalizar antes de compilar.

```
scss/_variables.scss


$focus-ring-width:       .25rem;
$focus-ring-opacity:     .25;
$focus-ring-color:       rgba($primary, $focus-ring-opacity);
$focus-ring-blur:        0;
$focus-ring-box-shadow: 0 0 $focus-ring-blur $focus-ring-width $focus-ring-color;
```

Luego, esas variables se reasignan a :rootvariables CSS de nivel que se pueden personalizar en tiempo real, incluso con opciones para xy ycompensaciones (que tienen como valor predeterminado su valor de reserva de 0).

```
scss/_root.scss


--#{$prefix}focus-ring-width: #{$focus-ring-width};
--#{$prefix}focus-ring-opacity: #{$focus-ring-opacity};
--#{$prefix}focus-ring-color: #{$focus-ring-color};
```

Puntos de interrupción de la cuadrícula

Si bien incluimos nuestros puntos de interrupción de cuadrícula como variables CSS (excepto xs), tenga en cuenta que las variables CSS no funcionan en consultas de medios . Esto es así por diseño en la especificación CSS para variables, pero puede cambiar en los próximos años con el soporte para env()variables. Consulte esta respuesta de Stack Overflow para obtener algunos enlaces útiles. Mientras tanto, puede usar estas variables en otras situaciones CSS, así como en su JavaScript.