

Custom CSS reference: selectors and examples

Design and templates

Use this reference to style your help center beyond the visual controls: restyle category cards, emphasize a navigation link, adjust article typography, or change a component at a specific screen size. It lists the selectors and theme variables built for customization, and ready-to-use examples.

It assumes you're comfortable reading CSS selectors and using your browser's developer tools. To learn where custom CSS lives and how to add a snippet, start with [Add custom CSS](#).

Who can do this: Owners and Admins (Editors through **Customize** on your help center) · **Plans:** All plans

Add and organize custom CSS

1. In the left menu, click **Template editor**, then open the **Advanced** tab.
2. Under **Custom CSS**, click **Add custom CSS**.
3. Give the snippet a descriptive label, such as Category cards.
4. Enter the CSS.
5. Click **Save** to fold the snippet.

6. Check the **Desktop**, **Tablet**, **Mobile**, **Light** and **Dark** previews.

7. Publish when the whole draft is ready.

Create a separate labeled snippet for each purpose. You can reorder snippets, disable one without deleting its code, or remove it. Enabled snippets are combined in the order shown, so a later rule overrides an earlier one when their specificity is equal.

The combined limit for all enabled snippets is 200 KB. Above it, none of your custom CSS is applied, and your draft can't be saved until you trim it.

How custom CSS is scoped

Every selector is scoped to `#hc-site`, the element that wraps your help center's pages. For example, this input:

```
.hc-article__title {  
  letter-spacing: -0.03em;  
}
```

is published as:

```
#hc-site .hc-article__title {  
  letter-spacing: -0.03em;  
}
```

You don't need to add `#hc-site` yourself. The extra scope also means a purpose-built `.hc-*` selector usually overrides the default styles without `!important`.

Custom CSS is cleaned the same way in the preview and on your published help center:

- `@import`, `@charset` and `@namespace` are removed, and so are comments.

- javascript: and HTML data: URLs are blocked, and behavior and expression() are removed.
- Unbalanced braces in any snippet turn off all your custom CSS, not only that snippet.
- Standalone html, body, * and :root selectors are rewritten to #hc-site.
- Selectors inside @media and @supports are supported and scoped automatically.
- @keyframes and @font-face keep their native structure.

Two consequences of the scope are worth knowing:

- #hc-site doesn't draw a box of its own. Inherited properties you set on html, body or :root, such as color, font-family or font-size, still reach the page. Backgrounds, borders and padding set there don't show. Set the page background under **Theme** instead.
- Selectors that start at the page root, such as html[data-theme="dark"] .hc-hero or body.some-class a, can't match, because #hc-site is added in front of them. For dark-mode styling, use the theme variables below. A rule inside @media (prefers-color-scheme: dark) also works, but it follows the reader's device setting, not Compass's light and dark switch.

To change fonts, pick them under **Theme** rather than with an @import rule. For a font that isn't in the list, an @font-face rule that points to a font file you host still works: declare it, then use it in a font-family rule.

Choose selectors that are designed for customization

Prefer selectors in this order:

1. data-hc-region for a page region

2. data-hc-component-type for every instance of a component
3. A purpose-built .hc-* class for a stable part of a component
4. data-hc-instance-id when only one component instance should change
5. A custom class assigned to a navigation link

Avoid generated utility classes, deeply nested element paths and selectors based on visible text. Those details are more likely to change as the default design evolves.

Region selector reference

Use region selectors when the rule should affect the entire band or column.

Area	Recommended selector
Header	[data-hc-region="header"]
Hero	[data-hc-region="hero"]
Main content	[data-hc-region="main"]
Sidebar	[data-hc-region="sidebar"]

Utility rail or band	[data-hc-region="utility"]
----------------------	----------------------------

Footer	[data-hc-region="footer"]
--------	---------------------------

Useful region classes include:

Element	Selector
Header band	.hc-region-header
Block header row	.hc-header-row--block
Inline header row	.hc-header-row--inline
Hero band	.hc-region-hero
Hero component	.hc-hero
Hero content	.hc-hero-content

Brand and navigation	[data-hc-component-type="Logo"], [data-hc-component-type="Navigation"], [data-hc-component-type="LanguageSwitcher"]
Hero and search	[data-hc-component-type="Hero"], [data-hc-component-type="SearchBar"]
Categories and discovery	[data-hc-component-type="CategoryGrid"], [data-hc-component-type="ArticleList"], [data-hc-component-type="FeaturedArticles"], [data-hc-component-type="Faqs"], [data-hc-component-type="ContentTree"]
Article information	[data-hc-component-type="ArticleMetadata"], [data-hc-component-type="TableOfContents"], [data-hc-component-type="RelatedArticles"], [data-hc-component-type="RecentlyViewed"]
Article actions	[data-hc-component-type="ArticleRating"], [data-hc-component-type="ShareButtons"], [data-hc-component-type="ArticleExport"], [data-hc-component-type="ArticleComments"]
Forms and calls to action	[data-hc-component-type="ContactForm"], [data-hc-component-type="ContactCta"]

Layout and content	[data-hc-component-type="Container"], [data-hc-component-type="Spacer"], [data-hc-component-type="RichText"], [data-hc-component-type="CustomHtml"]
Footer	[data-hc-component-type="Footer"]

A few simple components have no component hook. Use `.hc-breadcrumbs` for Breadcrumbs, and a surrounding region selector for Divider and Social links.

The built-in article listings on the Category and Search pages reuse the Article list markup but aren't component instances. Target them with `[data-hc-region="main"] section[aria-labelledby="hc-articles-heading"]:not([data-hc-component-type])`.

Element selector reference

Categories and article listings

Element	Selector
Category grid wrapper	.hc-card-list
Card grid	.hc-category-grid
Card-style grid	.hc-category-grid--cards

One category	[data-hc-category-id]
Category header in "Cards + top articles"	.hc-cat-head
Articles inside a category card	.hc-cat-articles
"See all" category link	.hc-cat-seeall
Parent-category sections	.hc-category-sections, .hc-category-section
Subcategory collection	.hc-subcats
One subcategory card	.hc-subcat
Subcategory title	.hc-subcat__title
An article link in search results	a[data-hc-article-id]

Article pages

The article classes below are stable custom CSS hooks.

Element	Selector
Article wrapper	.hc-article
Article header	.hc-article__header
Category eyebrow	.hc-article__eyebrow
Article title	.hc-article__title
Article body	.hc-article__content
Metadata row	.hc-article-meta
Author	.hc-article-meta__author
Updated date	.hc-article-meta__date

Reading time	.hc-article-meta__reading-time
Metadata separator	.hc-article-meta__sep
Breadcrumbs	.hc-breadcrumbs, .hc-breadcrumbs__item, .hc-breadcrumbs__current
Rating component	.hc-article-rating
Rating buttons	.hc-article-rating__button
Comments	.hc-article-comments
Export menu	.hc-article-export
Share buttons	.hc-share-buttons
Table of contents	.hc-toc, .hc-toc__menu

Search, forms and navigation

Element	Selector
Search submit button	.hc-search-submit
Ask AI button	.hc-ask-ai
Instant-search dropdown	.hc-search-dropdown
Instant-search results list	.hc-search-results
Search-bar AI answer	.hc-searchbar-ai
Contact form	#hc-contact-form
Article suggestions	.hc-contact-suggest
Contact CTA	.hc-contact-cta

Contact CTA button	.hc-contact-cta__btn
Content tree	.hc-content-tree
Current content-tree item	.hc-content-tree__item.is-current, .hc-content-tree__article.is-current
Language dropdown	.hc-lang-switcher
Inline language links	.hc-lang-inline
Logo image	.hc-logo-img
Light-mode logo image	.hc-logo-img--light
Dark-mode logo image	.hc-logo-img--dark
Formatted article or rich text	.hc-prose

Theme variable reference

Use theme variables instead of fixed colors where you can. They follow your **Light mode** and **Dark mode** palettes automatically, including a reader's manual switch between the two.

Purpose	Variable
Primary brand color	--hc-color-primary
Accent color	--hc-color-accent
Main text	--hc-color-fg
Page background	--hc-color-bg
Muted text	--hc-color-muted
Raised surface or card	--hc-color-surface

Borders	--hc-color-border
Primary or accent used as text, adjusted to stay readable	--hc-color-primary-text, --hc-color-accent-text
A button filled with the primary color, and the readable label color on it	--hc-color-primary-fill, --hc-color-on-primary
Form field outline, adjusted to stay visible	--hc-color-field-border
Automatic Hero text	--hc-hero-fg
Small, medium and large radius	--hc-radius-sm, --hc-radius-md, --hc-radius-lg
Pill radius	--hc-radius-pill
Body, heading and code fonts	--hc-font-body, --hc-font-heading, --hc-font-mono

Theme spacing

--hc-density-gap, --hc-density-pad

Theme H1 size

--hc-density-h1

Example:

```
.hc-contact-cta {  
  background: var(--hc-color-surface);  
  border: 1px solid var(--hc-color-border);  
  border-radius: var(--hc-radius-lg);  
  color: var(--hc-color-fg);  
}
```

Example: restyle the category boxes

This example gives vertical and horizontal category cards a quieter border, a stronger shadow and a lifted hover state:

```
[data-hc-component-type="CategoryGrid"] .hc-category-grid--cards > li > a {  
  background: var(--hc-color-bg);  
  border: 1px solid var(--hc-color-border);  
  border-radius: calc(var(--hc-radius-lg) + 0.25rem);  
  box-shadow: 0 2px 8px rgb(15 30 60 / 8%);  
  transition:  
    border-color 150ms ease,  
    box-shadow 150ms ease,  
    transform 150ms ease;  
}  
  
[data-hc-component-type="CategoryGrid"] .hc-category-grid--cards > li > a:hover {  
  border-color: var(--hc-color-primary);  
}
```

```

    box-shadow: 0 10px 28px rgb(15 30 60 / 14%);
    transform: translateY(-3px);
  }

  @media (prefers-reduced-motion: reduce) {
    [data-hc-component-type="CategoryGrid"] .hc-category-grid--cards > li > a {
      transition: none;
    }

    [data-hc-component-type="CategoryGrid"] .hc-category-grid--cards > li > a {
      transform: none;
    }
  }
}

```

Because it uses theme variables, the cards work in both light and dark mode. The prefers-reduced-motion block keeps the cards still for readers who turn off animations on their device.

Example: turn category icons into tiles

Font icons and uploaded images use different elements. Style both when your categories can use either:

```

[data-hc-component-type="CategoryGrid"] .hc-category-grid--cards > li > a > i {
[data-hc-component-type="CategoryGrid"] .hc-category-grid--cards > li > a > i {
[data-hc-component-type="CategoryGrid"] .hc-category-grid--cards > li > a > s {
[data-hc-component-type="CategoryGrid"] .hc-category-grid--cards > li > a > s {
[data-hc-component-type="CategoryGrid"] .hc-cat-head > i,
[data-hc-component-type="CategoryGrid"] .hc-cat-head > img {
  width: 3.25rem;
  height: 3.25rem;
  border-radius: var(--hc-radius-md);
}

[data-hc-component-type="CategoryGrid"] .hc-category-grid--cards > li > a > i {
[data-hc-component-type="CategoryGrid"] .hc-category-grid--cards > li > a > s {
[data-hc-component-type="CategoryGrid"] .hc-cat-head > i {
  display: inline-flex;
  align-items: center;

```

```

    justify-content: center;
    background: color-mix(in srgb, var(--hc-color-primary) 12%, transparent);
    color: var(--hc-color-primary);
    font-size: 1.25rem;
}

[data-hc-component-type="CategoryGrid"] .hc-category-grid--cards > li > a >
[data-hc-component-type="CategoryGrid"] .hc-category-grid--cards > li > a >
[data-hc-component-type="CategoryGrid"] .hc-cat-head > img {
    object-fit: contain;
    padding: 0.5rem;
    background: var(--hc-color-surface);
}

```

The icons become consistent branded tiles without changing the category content or the card layout. The `a > i` and `a > img` selectors cover vertical cards, `a > span > i` and `a > span > img` cover horizontal cards, and `.hc-cat-head` covers "Cards + top articles".

Example: change category spacing on smaller screens

```

@media (max-width: 640px) {
    [data-hc-component-type="CategoryGrid"] .hc-category-grid {
        gap: 0.875rem;
    }

    [data-hc-component-type="CategoryGrid"] .hc-category-grid--cards > li > a {
        padding: 1rem;
    }
}

```

This gives phone-sized screens more compact cards with less space between them. Check media-query changes in the **Mobile** preview and on a real phone before you publish.

Example: style the article title

```
.hc-article__title {  
  max-width: 22ch;  
  font-size: clamp(2.25rem, 6vw, 4.5rem);  
  line-height: 1.02;  
  letter-spacing: -0.04em;  
}
```

The clamp() value keeps the title responsive without separate desktop and mobile rules.

Example: hide the author site-wide

Use the component's settings when a single Article metadata component should change. Use CSS when the author must be hidden across every article layout:

```
.hc-article-meta__author,  
.hc-article-meta__author + .hc-article-meta__sep {  
  display: none;  
}
```

The second selector removes the separator that follows the author.

Example: customize the search button

```
.hc-search-submit {  
  background: var(--hc-color-accent);  
  border-radius: var(--hc-radius-pill);  
  padding-inline: 1.25rem;  
}  
  
.hc-search-submit:hover {  
  filter: brightness(1.08);  
}
```

When you change a button's background, check that its label is still easy to read in both light and dark mode. Keep a visible focus style too: don't remove outlines unless you replace them with an equally clear keyboard-focus treatment.

Example: emphasize one navigation link

In the **Navigation** tab, type a CSS class such as `support-cta` in the link's **CSS class (advanced)** field. Then target that class:

```
[data-hc-component-type="Navigation"] .support-cta {
  display: inline-flex;
  align-items: center;
  border-radius: var(--hc-radius-pill);
  background: var(--hc-color-primary-fill, var(--hc-color-primary));
  color: var(--hc-color-on-primary, #fff);
  padding: 0.5rem 1rem;
}

[data-hc-component-type="Navigation"] .support-cta:hover {
  color: var(--hc-color-on-primary, #fff);
  opacity: 0.88;
}
```

This is more durable than selecting a link by its URL, position or label. `--hc-color-on-primary` is black or white, whichever reads better on your primary color in each mode. The class is added to the link in Navigation components; the footer's default link list doesn't carry it.

Target one component instance

Most components carry a `data-hc-instance-id` value (Breadcrumbs, Divider and Social links don't). Use it when two instances of the same component need different styling:

```
[data-hc-instance-id="YOUR_INSTANCE_ID"] {
  max-width: 48rem;
  margin-inline: auto;
}
```

```
}
```

Find the value with your browser's developer tools and confirm it's on the rendered component. An instance ID belongs to that component: deleting and re-adding the component creates a different one. Prefer component-type selectors when every instance should share the rule.

Troubleshoot a rule that doesn't apply

1. Check that you published, and that the snippet is enabled.
2. Check that every opening brace has a closing brace. One unmatched brace turns off all your custom CSS.
3. Inspect the element and confirm the selector exists on the current component variant.
4. Don't start selectors at `html`, `body` or `:root`: custom CSS is scoped inside `#hc-site`.
5. Replace fixed colors with the `--hc-*` theme variables and check the **Dark** preview.
6. Check whether a later snippet overrides the rule.
7. Increase specificity by adding a region or component hook before you reach for `!important`.
8. Test hover, focus, open, selected, empty and mobile states.
9. Publish only after checking representative Home, Category, Article, Search, Contact and Not found pages.

Applying a theme with **Custom CSS** ticked replaces your snippets with the theme's. Untick it in the **Apply theme** dialog to keep yours, and review your snippets after any

theme change: the theme's CSS and your overrides may target the same hooks. For more fixes, see [My custom CSS doesn't work](#).

Related articles

- [Add custom CSS](#)
- [My custom CSS doesn't work](#)
- [Add links to your header and footer](#)
- [Light and dark mode](#)