Template Blueprint for Designers

This document describes how to create custom themes for Label Site Generator. You can override templates and CSS without modifying the generator code.

How Custom Themes Work

1. Place a custom `style.css` in `content/global/style.css` - it replaces the default CSS entirely
2. Override any `.njk` template in `templates/` - the generator reads from this directory
3. Both are gitignored (label-specific) and survive updates

File Structure

```
templates/
  base.njk          # Shared layout: head, hero, nav, footer, scripts
  index.njk         # Homepage: artists, releases, news, events, about
  artist.njk        # Artist page: hero, bio, gallery, events, discography
  album.njk         # Album/release page: artwork, streaming, physical, tracklist
  releases.njk      # Full release listing
  news-list.njk     # News listing with pagination
  news-article.njk  # Individual news article
  page.njk          # Static pages (imprint, contact, custom)

content/global/
  style.css         # Custom CSS (replaces default theme entirely)
```

Template Blocks (base.njk)

Override these in child templates:

Block	Purpose
`title`	Page `<title>`
`metaDescription`	Meta description
`ogType`, `ogTitle`, `ogDescription`, `ogImage`	Open Graph
`twitterTitle`, `twitterDescription`, `twitterImage`	Twitter Card
`jsonLd`	JSON-LD structured data
`hero`	Hero section (banner + logo)
`content`	Main page content

Available Variables (baseCtx)

These are available in every template:

Site Identity

Variable	Type	Source
`labelName`	string	`SITE_NAME` or `LABEL_NAME`
`siteUrl`	string	`SITE_URL` (with trailing slash)
`labelEmail`	string	`SITE_EMAIL`
`labelAddress`	string	`SITE_ADDRESS`
`labelVatId`	string	`SITE_VAT_ID`
`siteTagline`	string	`SITE_TAGLINE`
`labelBandcampUrl`	string	`BANDCAMP_URL`
`currentYear`	number	Current year

Content Flags

Variable	Type	Description
`hasBanner`	boolean	`assets/banner.jpg` exists
`hasLogo`	boolean	`assets/logo-round.png` exists
`hasNews`	boolean	News articles available
`hasEvents`	boolean	Upcoming events available
`showOtherLabels`	boolean	`OTHER_LABEL_CONTENT=true`

Data Collections

Variable	Type	Description
`artists`	array	All artists (sorted A-Z, no Various Artists)
`latestReleases`	array	Latest 12 releases for homepage
`totalReleases`	number	Total release count
`newsArticles`	array	Latest 10 news articles
`totalNews`	number	Total news count
`allEvents`	array	Next 10 upcoming events
`newsletter`	object	Newsletter config (provider, actionUrl, etc.)
`social`	object	Social media URLs
`extraPages`	array	Custom pages from content/pages/
`mainNavPages`	array	Pages with menu: main
`footerNavPages`	array	Pages for footer nav

Social Object

```
social.bandcamp
social.spotify
social.soundcloud
social.youtube
social.instagram
social.facebook
social.tiktok
social.twitter
```

Newsletter Object

```
newsletter.provider    # "sendy", "listmonk", "keila", or ""
newsletter.actionUrl   # Server URL
newsletter.listId      # List identifier
newsletter.formId      # Keila form ID
newsletter.doubleOptIn # boolean
```

Artist Object

Available in `artists` array and as `artist` on artist/album pages:

```
artist.name           # Display name
artist.slug           # URL slug
artist.location       # City, Country (or null)
artist.description    # Bio text from Bandcamp
artist.photo          # Photo filename (or null)
artist.coverImage     # Cover image URL
artist.topTags        # Array of top 3 genre tags (normalized)
artist.albums         # Array of album objects
artist.events         # Array of event objects
artist.streamingLinks # { spotify, appleMusic, bandcamp, ... }
artist.bandLinks      # Array of { name, url }
artist.eventLinks     # { bandsintown, songkick }
```

Album Object

Available in `latestReleases`, `allAlbums`, and as `album` on album pages:

```
album.title           # Album title
album.slug            # URL slug
album.url             # Bandcamp URL (null for Spotify-only)
album.artwork         # Artwork path or URL
album.releaseDate     # ISO date string
album.description     # Album description
album.credits         # Credits text
album.tags            # Array of { name } tag objects
album.tracks          # Array of { name, duration, isrc }
album.videos          # Array of { url, title }
```

```
album.streamingLinks  # { spotify, appleMusic, deezer, tidal, ... }
album.physicalFormats # ["Vinyl", "CD", "Cassette"]
album.labelName       # Label name (from Spotify)
album.labelUrl        # Discogs label URL
album.discogsLabel    # Physical release label (if different)
album.upc             # UPC barcode
album.catalogNumber   # Catalog number
album.upcoming        # boolean
album.tier            # "announce", "preview", "full"
album.presaveUrl      # Pre-save link (or null)
album.privateUrl      # Private Bandcamp URL (or null)
album.albumId         # Bandcamp album ID (for embed player)
album.itemType        # "album" or "track"

# On homepage/releases (added by renderer):
album.artistName      # Artist display name
album.artistSlug      # Artist URL slug
```

Event Object

```
event.date            # ISO date string
event.name            # Event/festival name
event.venueName       # Venue name
event.cityName        # City
event.countryName     # Country
event.type            # "concert" or "festival"
event.eventUrl        # Ticket/event URL
event.artistName      # Artist name (homepage events)
event.artistSlug      # Artist slug (homepage events)
```

News Article Object

```
article.title         # Article title
article.slug          # URL slug
article.date          # ISO date string
article.excerpt       # Short excerpt
article.image         # Feature image path/URL
article.imageUrl      # Resolved image URL
article.html          # Full HTML content (Ghost) or rendered markdown
```

Custom Filters

Available in all templates:

Filter	Usage	Description
`formatDate`	`{{ date | formatDate }}`	"1 November 2026"
`isFuture`	`{{ date | isFuture }}`	true if date is in the future

`isLocal`	`{{ url | isLocal }}`	true if not http/https
`toWebp`	`{{ img | toWebp }}`	.jpg/.png to .webp
`toMobileWebp`	`{{ img | toMobileWebp }}`	.jpg to -mobile.webp
`urlencode`	`{{ str | urlencode }}`	URL-encode
`nl2br`	`{{ text | nl2br }}`	Newlines to ` `
`youtubeId`	`{{ url | youtubeId }}`	Extract YouTube video ID
`availableFormats`	`{{ album | availableFormats }}`	"Vinyl, CD, Digital - Label"
`storeUrl`	`{{ tpl | storeUrl(artist, album) }}`	Replace {artist}/{album}

CSS Variables (Default Theme)

Override these in `content/global/style.css`:

```
:root {
  --brand-dark: #0c0032;
  --brand-light: #cacadb;
  --brand-mid: #3a3a6e;
  --brand-accent: #5a5a9e;
  --bg: #f7f7fa;
  --surface: #ffffff;
  --border: #dcdce8;
  --text: #0c0032;
  --text-muted: #6b6b8a;
  --accent: #0c0032;
  --accent-hover: #3a3a6e;
  --header-bg: #0c0032;
  --header-border: #1e1e5a;
  --footer-bg: #0c0032;
  --max-width: 1200px;
}
```

When Bandcamp theme colors are detected, they override `--bg`, `--text`, `--accent`, etc. automatically. A custom `style.css` takes full precedence.

Minimal Custom Theme Example

```
/* content/global/style.css - Minimal dark theme */
*, *::before, *::after { box-sizing: border-box; margin: 0; padding: 0; }

:root {
  --bg: #1a1a2e;
  --surface: #16213e;
  --text: #e0e0e0;
  --accent: #e94560;
```

```
  --max-width: 1200px;
}

body { background: var(--bg); color: var(--text); font-family: system-ui, sans-serif; }
a { color: var(--accent); }

/* ... add your styles ... */
```

Tips

- Start by copying the default CSS from `src/assets.js` (the `DEFAULT_CSS` constant)
- Use browser DevTools to inspect class names on the live site
- All images have `loading="lazy"` - no need to handle lazy loading
- The `rootPath` variable handles relative URLs (use `{{ rootPath }}style.css`)
- Test with `npx serve dist` after `node generate.js`