LULU STEP-BY-STEP GUIDE

Theme package format reference

Every file and key in a theme package (shopwise-theme.json, config/groups.json, templates), its type and limit, which section types each presetKey offers, fonts, images, CSS hooks, custom.js, and what the format cannot do.

For: Everyone · About 25 minutes · 14 steps · Updated November 20, 2026

Before you start This is the reference for the files inside a theme package. For the step-by-step upload and submission flow, read “Edit a theme package and submit it to the Theme Store” first. Everything here comes from the platform code: anything the platform doesn't recognise is dropped on upload, never stored. The same tables are in your dev store's admin under Online Store → Developer docs, generated live from the code.

Steps at a glance

1. The package and what is read back
2. Add a package from GitHub
3. shopwise-theme.json: top-level keys
4. settings: design settings
5. sections: page layouts
6. config/groups.json: header, footer, announcement bar and password page
7. Add product, collection and page templates
8. presetKey: the base theme and its section palette
9. Fonts
10. Images
11. CSS: theme-wide and per section
12. custom.js and head snippets
13. What a theme can't do
14. Updates after merchants install your theme

STEP 1

The package and what is read back

A package is a zip (5 MB at most) or the shopwise-theme.json file on its own. On upload, Lulu reads shopwise-theme.json plus the copies of the same data you edited: config/settings_data.json, templates/*.json, config/groups.json and assets/custom.css. Each export stores a fingerprint of those copies in shopwise.derived, so an untouched copy changes nothing and an edited copy replaces the matching part of shopwise-theme.json. If you edit the same part in both places, shopwise-theme.json wins.

- Read back: shopwise-theme.json, config/settings_data.json (only its “current” object), templates/*.json, config/groups.json, assets/custom.css.
- Never read back: config/settings_schema.json, sections/*.json, README.md, README.txt, assets/custom.js and snippets/head-snippets.html. Custom JavaScript and head snippets come only from codeFiles in shopwise-theme.json.
- shopwise-theme.json can sit at the root of the zip or one folder down. Two copies at the same depth are refused.
- A package downloaded before 9 October 2026 has no fingerprints: only shopwise-theme.json is used; edited copies are ignored and the upload names them.
- Re-formatting a file (spacing, key order) is not an edit. An emptied template file is ignored: hide sections with "disabled": true instead.
- Every value goes through the same checks as a save in the theme editor, so a package can't store anything the editor couldn't.

STEP 2

Add a package from GitHub

Instead of uploading a zip, you can keep the package in a GitHub repository. In your store admin, go to Online Store → Import and choose Add from GitHub.

- Repository URL: https://github.com/owner/repo, or just owner/repo. Only repositories on github.com can be added.
- Branch: leave it blank to use main.
- Subfolder: the folder that holds shopwise-theme.json, for example themes/aurora. Leave it blank when shopwise-theme.json is at the top of the repository.
- Personal access token: only needed for a private repository. Create a fine-grained token with read-only access to that repository's Contents. It is used for that one download and never saved, so you paste it again each time you sync.
- Only the package's text files are read (shopwise-theme.json, config/, templates/, assets/custom.css and the like). Images, fonts and other files in the repository are skipped. The download can be up to 50 MB; the package files in it have the same 5 MB limit as a zip upload.
- The theme is added as a draft with a GitHub label. To pull the latest commit later, open the theme's … menu and choose Sync from GitHub. Sync replaces the theme's settings, sections and code with the files on GitHub, and saves what was there before in the theme editor's History so you can roll back. Syncing the live theme asks you to confirm first, because customers see the change right away.
- Disconnect from GitHub, in the same menu, keeps the theme as it is and stops syncing.

STEP 3

shopwise-theme.json: top-level keys

The main file is a JSON object. A {"snapshot": {…}} wrapper around the same keys is also accepted.

- name (text): the theme name, up to 255 characters. If the store already has a theme with that name, “(imported)” is added.
- settings (object): design settings, key = setting name, value = text. Only the 55 known settings are kept; an unknown name or a value outside its range is dropped. See “settings”.
- customCss (text): theme-wide CSS, up to 20,000 characters. A literal </style> is escaped; nothing else is changed.
- sections (object or null): every page layout plus the header, footer and password page. null means the stock home page layout and empty templates. See “sections”.
- codeFiles (object): {"customJs": text up to 100,000 characters, "headSnippets": text up to 20,000 characters}. Other keys are dropped.
- presetKey (text): the base theme your package builds on. It picks the default colours and fonts and which section types the editor offers. See “presetKey”.
- shopwise (object): format information. version (number, must be 1: it is the file format, not your theme's version), themeVersion (your version, 1 to 4 dot-separated numbers such as 2.0.1, up to 20 characters; anything else is ignored with a warning), kind, exportedAt, store, themeId, name, source, presetKey (a copy for reading; the top-level presetKey is the one used), provenance {catalogKey, listingId} and derived (the file fingerprints, don't edit them).

STEP 4

settings: design settings

settings holds design tokens such as "--s-accent": "#0a5c46". There are 55 of them, in the groups Colors, Typography, Layout, Buttons, Cards, Forms, Icons, Animations and Responsive; config/settings_schema.json in your package and Online Store → Developer docs list each one with its type and range.

- color: #RGB, #RRGGBB or #RRGGBBAA.
- size: whole pixels such as "14px", within the setting's minimum and maximum.
- percent: such as "80%", within the setting's range.
- select: one of the setting's listed values.
- font: a font id from the font list (see “Fonts”). "theme" is not stored: leave the setting out to use the preset's own font.
- A setting you leave out uses the preset's default. In config/settings_data.json, “current” replaces all settings, so deleting a key there also resets it.

STEP 5

sections: page layouts

sections has these keys. Each section is {"id", "type", "settings", "blocks"?, "hidden"?}; ids are 1–40 letters, digits, _ or - (anything else gets a new random id). Blocks are {"id", "type", "settings", "hidden"?, "blocks"?} and nest at most 3 levels, 80 blocks per section in total.

- home (list, required when sections is not null): home page sections, up to 40, of any registered section type.
- templates (object): product, collection, cart, blog (blog list and articles), search and page, each a list of up to 15 sections. Any type except announcement-bar. A template section with "area": "top" renders above the page's own content (in templates/*.json that is "shopwise_area": "top").
- pageTemplates (object): alternate page templates, key = suffix such as "contact" (lowercase letters, digits and -, up to 40 characters), up to 20 of them, each a list like templates.page.
- resourceTemplates (object): {"product": {suffix: list}, "collection": {…}, "post": {…}} — alternate product, collection and blog-article templates. Suffix: lowercase letters, digits and -, up to 25 characters; up to 20 per type.
- templateCore (object): settings of each template's built-in layout (product gallery and buy box, collection grid, cart, blog, search, page) — keyed product, collection, cart, blog, search, page. It is only in shopwise-theme.json, not in templates/*.json.
- headerGroup, footerGroup, passwordPage: see the next step. They are also in config/groups.json.
- Unknown section types and setting keys are left out and listed after the upload (on the submission page, or on the Themes page after Online Store → Import). The section types that exist are the sections/ folder of any package and the Section library in Developer docs.

STEP 6

config/groups.json: header, footer, announcement bar and password page

config/groups.json is {"headerGroup", "footerGroup", "passwordPage"}. A group you include replaces that whole group; a group you set to null is left as it is in shopwise-theme.json.

- headerGroup.settings: 95 header settings (logo, menu, search, icons, sticky, transparent header, colours, customCss). headerGroup.blocks: up to 6 blocks of type text, icon, image, button, custom-html, shown beside the header icons.
- Header layout and icons, so each theme can have its own header: menuRow top | bottom | above (above = the menu on the first row, logo and icons on the row under it), logoPosition left | center, menuPosition left | center | right, utilitiesPosition right | split (split = search icon at the left, the other icons on the right), accountIcon person | circle | round | square, cartIcon bag | cart | tote | basket | cart-outline, wishlistIcon heart | bookmark | star, searchIcon magnifier | text, iconSize small | standard | large. Unknown values fall back to the default.
- headerGroup.announcementBar: {"settings", "blocks"}. Up to 10 blocks: announcement (text, textMobile, link, newTab, scheme, startAt, endAt; startAt/endAt schedule it) and utility-link (role, label, link, newTab, icon, iconOnly). headerGroup.announcement {enabled, text, link} is the older single-message bar, used only while announcementBar has no blocks.
- footerGroup.sections: up to 6 sections above the footer, of type newsletter, rich-text, trust-badges, brands, multicolumn, text, icons-with-text, divider, marquee, contact-form, contact-info, location-map, offer-popup, stickers, scroll-to-top, age-verifier, popup.
- footerGroup.settings: the classic footer (menus, newsletter, social icons, payment icons, copyright, columns, colours, customCss). footerGroup.blocks: up to 8 social links (platform, url, label, iconImage).
- footerGroup.layoutBlocks: the block-built footer. Empty or missing means the classic settings footer is used. Up to 24 top-level blocks and 80 blocks in total. Types: group, heading, text, image, logo, button, menu, links, social, email-signup, payment-icons, copyright, policy-links, divider, jumbo-text. A group holds up to 12 children of any of those types (groups nest up to 3 levels); a links block holds up to 8 link children.
- passwordPage.settings: heading, message, backgroundImage, logoImage, passwordLabel, buttonLabel.

STEP 7

Add product, collection and page templates

To add a template, add a file to templates/ and upload the package. A file you add counts as an edit, so it is used.

- Shared templates: templates/product.json, collection.json, cart.json, blog.json, search.json, page.json. Home is templates/index.json.
- Alternates: templates/page.<suffix>.json (for example page.contact.json), templates/product.<suffix>.json, templates/collection.<suffix>.json and templates/article.<suffix>.json (stored as resourceTemplates.post).
- File shape: {"sections": {"<id>": {"type", "settings", "blocks"?, "block_order"?, "disabled"?}}, "order": ["<id>", ...]}. Copy an exported file as a starting point.
- A template file with no sections is ignored. You can't remove a template by uploading a package: delete it in the theme editor.
- A page, product, collection or article uses an alternate template once the merchant picks it under Theme template in its editor. That list shows the templates of the LIVE theme.

STEP 8

presetKey: the base theme and its section palette

presetKey must be the id of a registered preset. It decides the default colours and fonts underneath your settings, and which section types the editor's “Add section” list offers. It does not remove sections: any registered section type in your package is kept and keeps rendering even if the preset doesn't offer it. An unknown presetKey is ignored and the upload warns you, listing the valid keys; the theme then gets the default (Horizon) colours and fonts unless its name matches a preset, and the full section list.

- Core sections, offered by every preset: announcement-bar, newsletter, offer-popup, popup, stickers, scroll-to-top, age-verifier, hero, image-banner, image-with-text, video, slideshow, gallery, image-hotspots, featured-collection, collection-list, featured-product, recommended-products, product-highlight, product-hotspots, bundle-offer, product-compare, specifications, rich-text, faq, multicolumn, text, divider, icons-with-text, contact-form, contact-info, location-map, blog-posts, collapsed-rows, testimonials, trust-badges, brands, custom-html.
- Always offered as well (app sections): app-block, presale-banner, bb-canvas, ad-slot, foryou-row.
- Presentational sections that differ per preset: ribbon-banner, content-carousel, editorial, video-banner, video-with-text, pull-quote, marquee, jumbo-text, image-compare, stats-counter, section-tabs, testimonial.
- Full library (all 56 section types, including marketplace-discovery): horizon, ignite, vertex, splendor, atelier, circuit. Every paid Theme Store preset key, and a missing or unknown presetKey, also get the full library.
- dawn (Dawn): core + content-carousel, editorial, image-compare, stats-counter, section-tabs, video-banner, video-with-text.
- craft (Craft): core + editorial, pull-quote, content-carousel, image-compare, marquee, video-with-text.
- sense (Sense): core + editorial, content-carousel, image-compare, stats-counter, section-tabs, video-with-text.
- refresh (Refresh): core + jumbo-text, marquee, ribbon-banner, video-banner, content-carousel, pull-quote.
- studio (Studio): core + editorial, pull-quote, content-carousel, image-compare, section-tabs, video-with-text.
- colorblock (Colorblock): core + marquee, jumbo-text, ribbon-banner, video-banner, content-carousel, stats-counter.
- publisher (Publisher): core + editorial, pull-quote, jumbo-text, content-carousel, section-tabs, marquee.
- crave (Crave): core + ribbon-banner, marquee, video-banner, video-with-text, content-carousel, stats-counter.
- flow (Flow): core + editorial, marquee, content-carousel, image-compare, stats-counter, video-with-text.
- voyage (Voyage): core + editorial, pull-quote, content-carousel, image-compare, video-with-text.

- impulse (Impulse): core + jumbo-text, marquee, ribbon-banner, video-banner, content-carousel, pull-quote.
- instance (Instance): core + content-carousel, editorial, stats-counter, section-tabs, ribbon-banner, video-with-text.
- satin (Satin): core + editorial, pull-quote, content-carousel, image-compare, video-banner.
- forest (Forest): core + editorial, pull-quote, content-carousel, image-compare, marquee, video-with-text.
- composition (Composition): core + jumbo-text, marquee, editorial, content-carousel, stats-counter, video-banner.
- pristine (Pristine): core + editorial, content-carousel, image-compare, stats-counter, video-with-text.
- loft (Loft): core + marquee, ribbon-banner, jumbo-text, content-carousel, editorial, video-with-text.
- noom (Noom): core + editorial, pull-quote, content-carousel, video-banner, video-with-text, image-compare.
- igloo (Igloo): core + marquee, ribbon-banner, content-carousel, stats-counter, video-banner, image-compare.
- Paid Theme Store preset keys (for example prestige) are valid too, but they can mark your draft as that paid theme, and publishing it then needs the purchase. Use a free preset key for your own theme: horizon, vertex, dawn, craft, sense, refresh, studio, colorblock, publisher, crave, flow, voyage, impulse, instance, satin, forest, composition, pristine, loft, noom, igloo. ignite, splendor, atelier and circuit exist for existing themes and paid listings.
- Changing presetKey (for example crave to composition) changes both the section palette and the default look; set colours and fonts in settings if you want them fixed.

STEP 9

Fonts

The Heading font and Body font settings take one of 12 catalog fonts: system (System sans), inter (Inter), poppins (Poppins), sora (Sora), space-grotesk (Space Grotesk), assistant (Assistant), nunito (Nunito), lato (Lato), fraunces (Fraunces (serif)), cormorant (Cormorant (serif)), spectral (Spectral (serif)), georgia (Georgia (serif)). A theme can't add fonts to that list: the list is part of the platform code.

- Weights each catalog font loads: Inter, Poppins and Fraunces 400 to 900; Nunito 400 and 600 to 900; Sora, Assistant and Spectral 400 to 800; Lato 400, 700 and 900; Space Grotesk 400 to 700; Cormorant 500 to 700. A weight setting heavier than the font has (for example 800 on Space Grotesk) shows as its heaviest weight.
- The preset's own pair (the “Theme default” choice) can be a font outside the catalog, for example Barlow on Vertex or Figtree on Atelier. Leave the font settings out to use it.
- For another font, add a Google Fonts or Fontshare stylesheet link in head snippets (codeFiles.headSnippets) and set font-family in customCss. Other font hosts are blocked by the storefront's content security policy. Such a font never appears in the Typography picker, and because your CSS sets the font directly, a catalog font a merchant picks there won't replace it.

STEP 10

Images

A package never contains image files. Image settings (and url(...) in CSS) hold an address: a path on the store such as /media/stores/<id>/media/<file>, or any https:// address (up to 500 characters).

- Images uploaded to a store are only served on that store. Imported into another store with Online Store → Import, they don't show (they can still show to you while you are signed in to the store they came from).
- Theme Store packages: when you submit a theme, upload a package to your listing, or update the listing from a development store, images from your own dev stores are copied into the listing's package and the addresses are rewritten, so they show in the Theme Store demo and on every store that installs the theme. At most 200 images and 100 MB; JPG, PNG, WebP, GIF, AVIF, MP4 or WebM only. SVG is refused: the upload stops and names the file to replace.
- Images on a public https:// address are used as they are and show everywhere.
- Store content (products, collections, menus, blog posts, pages) is never in a package; an installed theme shows the installing store's own.

STEP 11

CSS: theme-wide and per section

customCss (20,000 characters) is added after the theme's own styles on every storefront page. Each section, the header, the footer and the announcement bar also have their own customCss setting of up to 4,000 characters.

- Per-section CSS is scoped: it is wrapped in that section's [data-sw-sec="<section id>"] selector with CSS nesting, so h2 { … } only styles that section's headings. Header, footer and announcement-bar CSS are scoped to [data-sw-hdr], [data-sw-footer] and [data-sw-ann] the same way.
- In per-section CSS, @import and expression(are removed, a stray } is dropped and unclosed rules are closed, so it can't escape its section.
- Stable hooks: every section carries .sw-<type> (for example .sw-rich-text) and every shared block carries .sw-block-<type> (for example .sw-block-heading). They come from the type name and are the documented classes to target.
- data-section-id="<id>" is on every section wrapper. The id stays the same across saves but a duplicated section gets a new one, so prefer .sw-<type>.
- Other classes, such as .s-header, .pc (product card) or .s-btn, are internal. Existing markup is kept stable on purpose, but these names are not a published contract and can change. Design tokens (var(--s-accent) and the other settings) are the safest thing to build on.

STEP 12

custom.js and head snippets

codeFiles.customJs runs once per full page load, at the end of <body>, on storefront pages only. Moving between storefront pages happens without a page reload and does not run it again, and there is no platform event after a section renders or a page changes.

- To react to new content, use a MutationObserver on document.body.
- Events you can listen for: sw:consent on window (cookie choice changed), sw:wishlist:change on document (Wishlist Plus), sw:quickview:open, sw:quickview:close and sw:quickview:add on document (Quick view).
- Tracking code (Google, Meta, TikTok and similar) waits for the visitor's cookie consent. Wrap other tracking in window.swPixel('analytics' or 'marketing', function () { … }).
- headSnippets is split into individual meta, link, base, script and style tags in <head>.

STEP 13

What a theme can't do

Themes are configuration for one shared storefront engine. A package can't contain Liquid, section source code or new section types, and Liquid support is not planned (it is an open owner decision, not on the roadmap).

- Your own markup: use the Custom HTML/Embed section (custom-html, offered by every preset, up to 20,000 characters) or the Custom HTML block in the header. The HTML renders as written, but <script> tags inside it don't run: put scripts in custom.js.
- Edit code in the admin offers custom.css, custom.js and head-snippets.html. settings.json and sections.json there are read-only views.
- No translation files, no layout files, no image files.

STEP 14

Updates after merchants install your theme

Installing a Theme Store theme creates a separate draft copy in the merchant's store. When you upload a new package or sync from your dev store, new installs get the new version. Copies already installed are never changed, so a merchant's own edits are never overwritten — and they don't receive your update either.

- There is no update notice or merge. A merchant who wants your new version installs the theme again (a new draft) and moves their changes across.
- Keep each released package and its themeVersion: the platform keeps no release history for a listing.

Check your work

- shopwise.version is 1 and themeVersion looks like 2.0.1.
- presetKey is a free preset id whose section list contains every section type you use.
- Fonts are catalog ids, or the preset's own pair.
- Images are on your dev store or on a public https:// address, none of them SVG.
- After uploading, the submission page lists the files you edited and no unexpected warnings.

Troubleshooting

A section disappeared after upload.

Its type isn't registered, so it was left out. Use a type from the sections/ folder of a downloaded package. A section type that exists but isn't in your preset's list is kept; it only doesn't appear in “Add section”.

My presetKey change did nothing.

An unknown presetKey is ignored, and the upload warns you with the list of valid keys. Use one of them. Remember that the top-level presetKey is used, not the copy inside the shopwise block.

Images are blank on another store.

They are links to images uploaded to the store the theme came from. Upload the images in the new store and choose them again, or use public https:// addresses. Theme Store submissions, listing uploads and listing updates from a development store copy dev-store images for you.

My edit to custom.js in the zip was ignored.

assets/custom.js and snippets/head-snippets.html are never read back. Edit codeFiles in shopwise-theme.json.

Add from GitHub says it couldn’t find the repository or branch.

Check the spelling of owner/repo and the branch name. A private repository needs a personal access token with read access to its Contents — without one, GitHub answers as if the repository didn’t exist. Tokens are never saved, so a private repository asks for one on every sync.

Need a hand? Contact support at https://lulu.us.com/contact. Screenshots use demonstration accounts; personal details were replaced.