Appendix A: Theme Creation Guide

Ensemble's appearance is controlled by a theme system that makes it straightforward to create a completely different look without touching any PHP code. A theme is a folder containing a CSS file. Drop it in the right place and it appears in the Theme Selection modal automatically.

This appendix is aimed at self-hosters and community contributors who want to build their own theme. You do not need to be a web developer — a basic colour-swap theme is just a handful of lines, and a complete working example is provided below.

A.1 What a Theme Is

A theme is a subdirectory of the `themes/` folder in your Ensemble installation. Any subdirectory that contains a `style.css` file is treated as a valid theme and appears in the Theme Selection modal immediately — no configuration, no code changes, no server restart required.

The theme system is entirely CSS-based. Ensemble's HTML structure, PHP logic, and JavaScript are unchanged by theme selection. A theme controls only how things look: colours, backgrounds, shadows, and optionally fonts and layout details.

A.2 Required Files

Each theme folder must contain these files:

File	Purpose
`style.css`	Required. The theme stylesheet.
`ensemblelogo.png`	Required. Logo shown in the page header.
`placeholder.png`	Required. Shown for outfits with no uploaded image.
`preview.png`	Optional. Small thumbnail shown in the Theme Selection modal (recommended size: 48×32 px).

The minimum viable theme folder looks like this:

```
themes/
   mytheme/
      style.css          ← required
      ensemblelogo.png   ← required (copy from themes/default/)
      placeholder.png    ← required (copy from themes/default/)
      preview.png        ← optional thumbnail
```

For a first theme, copy `ensemblelogo.png` and `placeholder.png` directly from `themes/default/`. You can replace them with custom versions later if you want.

A.3 The Simplest Possible Theme

Because Ensemble's entire UI is built on CSS custom properties (variables), a theme can consist of a single import line plus a handful of colour overrides. Here is a complete, working green-accent theme:

themes/green-forest/style.css

```
@import url('../default/style.css');  :root {     --color-primary:
#2e7d32;     --color-primary-hov: #1b5e20;     --color-border:
#c8e6c9;     --color-bg:          #f1f8f1;     --rose-100:
#f1f8f1;     --rose-200:          #c8e6c9; }
```

That is the entire file. The `@import` line loads the full default stylesheet first — all the layout, spacing, fonts, modal shapes, button styles, and structural CSS. The `:root` block then overrides only the specific colour tokens chosen. Every button, border, hover state, and card background in the interface will reflect the new colours because they all reference these variables.

Everything not overridden falls through to the default theme unchanged. You only need to specify what you want to be different.

> **Try it:** Create the folder, add these three files (two copied from `themes/default/`, one CSS file with the content above), and open the Theme Selection modal. Your theme appears immediately. Select it and click Apply — the whole interface switches to forest green without reloading.

A.4 How the Built-in Dark Theme Works

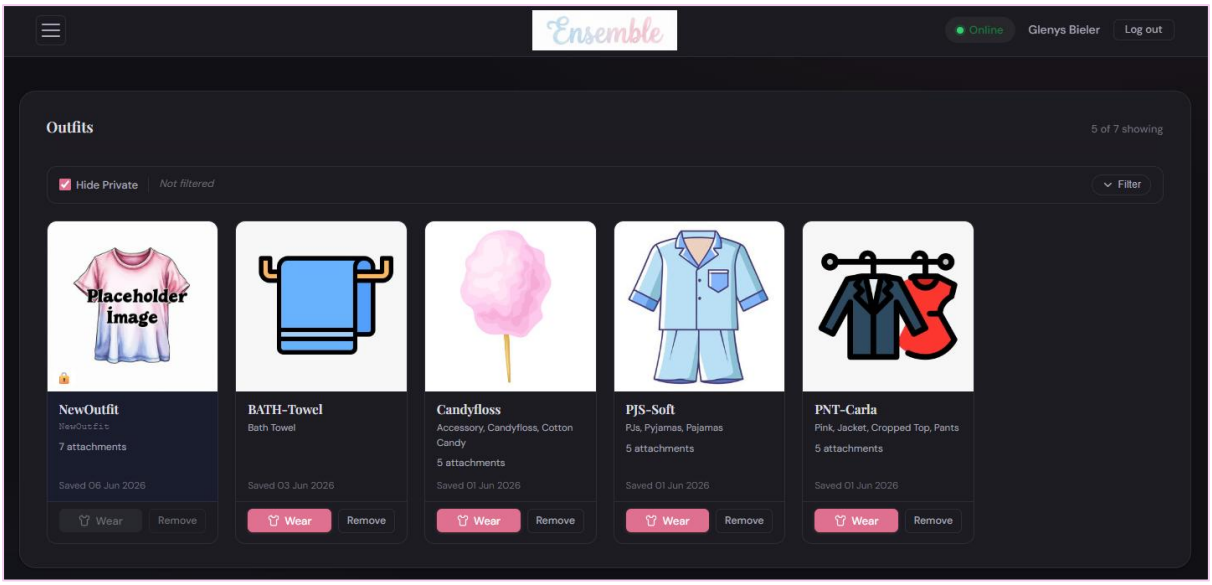
The dark theme that ships with Ensemble uses exactly this pattern, but overrides more tokens. Its entire `style.css` begins with:

themes/dark/style.css

```
@import url('../default/style.css');
```

It then overrides all the palette tokens (`--rose-*`, `--peri-*`, `--neutral-*`) and the semantic tokens (`--color-bg`, `--color-surface`, `--color-border`, `--color-text`, and so on) to their dark equivalents, and adjusts the shadow values to suit dark surfaces.

The result: structural changes to the default theme are automatically inherited by the dark theme — they stay in sync without duplication. This same pattern is available to any community theme.



The Ensemble web panel in Dark mode, showing the deep slate background, rose and violet accents, and outfit cards with dark surfaces

A.5 The Full CSS Token Reference

The following table lists every custom property defined in `themes/default/style.css` that a theme author can override. These are the theming API — they are stable and will be maintained across Ensemble releases.

CSS Custom Property	What it controls
Core palette tokens	
`--rose-100` … `--rose-600`	Rose/pink scale, lightest to darkest
`--peri-100` … `--peri-500`	Periwinkle/violet scale, lightest to darkest
`--neutral-50` … `--neutral-900`	Grey scale, lightest to darkest
Semantic tokens	
`--color-bg`	Page background colour
`--color-surface`	Cards, modals, and panels
`--color-border`	All borders throughout the UI
`--color-text`	Primary body text
`--color-text-muted`	Hints, placeholders, secondary text
`--color-primary`	Primary action colour (buttons, active states)
`--color-primary-hov`	Primary colour on hover
`--color-accent`	Secondary accent (chips, pickers, links)
Status colours	

CSS Custom Property	What it controls
`--color-online`	HUD Online status indicator
`--color-stale`	HUD Stale status indicator
`--color-offline`	HUD Offline status indicator
Layout and shape	
`--radius-sm` / `--radius-md` / `--radius-lg`	Border radius scale
`--shadow-sm` / `--shadow-md` / `--shadow-lg`	Box shadow scale
`--max-width`	Maximum page content width
`--header-h`	Header bar height

A theme that overrides only the semantic tokens (`--color-*` group) will cover the majority of the visible UI. Overriding the raw palette tokens (`--rose-*`, `--peri-*`) gives more complete coverage, since some components reference palette tokens directly rather than the semantic layer.

> **Where to look:** Open `themes/default/style.css` and find the `:root { }` block near the top. All the custom properties and their default values are listed there. Copying this block into your theme's `:root` and changing the values you care about is a reliable way to start.

A.6 Beyond Token Overrides

A theme can include any CSS rules it likes. It is not limited to `:root` variable overrides. Some things you can do with additional rules:

- **Change the page background gradient.** The default theme uses a subtle rose-tinted gradient on the page background. Override the `body` or `.page-wrapper` background property to replace it with a solid colour, a different gradient, or even an image.
- **Change fonts.** Load a different typeface with a `@import` or `@font-face` rule, then override `font-family` on `body` and the heading elements. The default theme uses Playfair Display for headings and DM Sans for body text.
- **Add a background pattern or texture.** A repeating SVG pattern or subtle noise texture applied to the page background can give a theme a distinctive feel.
- **Restyle specific components.** Every Ensemble UI component has class names you can target. For example, `.outfit-card`, `.side-menu`, `.header-bar`, `.modal-box`. Use your browser's developer tools to inspect any element and find the class name to target.
- **Override layout tokens.** Change `--max-width` to constrain or widen the content area, or `--header-h` to adjust the height of the header bar.

One rule: Always import `themes/default/style.css` first (the `@import url('../default/style.css')` line at the top of your file). This ensures all structural CSS is in place before your overrides run. If you omit the import, your theme is responsible for all CSS — including layout, fonts, modal structure, and every component — which is a much larger undertaking.

A.7 The Logo and Placeholder Images

ensemblelogo.png

This image is shown in the top-left of the header bar. On the default (light) theme it uses a white-background PNG. If your theme has a dark background, the default logo may look out of place.

Two approaches work well for dark-background themes:

- Use a logo image with a transparent background (PNG with alpha channel). This requires preparing a custom logo file.
- Use the default logo file but apply `mix-blend-mode: screen` in your CSS. This renders the white background of the PNG as transparent against dark surfaces — it is the technique the built-in dark theme uses:

```
In your theme's style.css — after the @import line
.logo-img {     mix-blend-mode: screen; }
```

`mix-blend-mode: screen` makes white pixels become transparent and coloured pixels become lighter. It works well on a dark background and means you can reuse the standard logo file without creating a custom one.

placeholder.png

This image is shown on outfit cards that have no uploaded image. It should be a recognisable "no image" graphic that looks appropriate against your theme's card background colour. For a first theme, copying the default placeholder is fine. Replace it with a custom image if the default does not sit well against your theme's colours.

A.8 The Preview Image

If `preview.png` exists in your theme folder, it is shown as a small thumbnail in the Theme Selection modal next to your theme's name. It is entirely optional — if absent, the thumbnail area is empty but the theme works normally.

Recommended content: a small crop of the interface showing your theme's colour palette. A good choice is the header bar and one or two outfit cards. Keep it representative — the thumbnail helps users choose between themes before applying one.

Recommended size: **48×32 pixels** or a proportional multiple (96×64, 192×128). The modal scales the image to fit the thumbnail slot, so exact dimensions are flexible, but a landscape crop at roughly 3:2 looks best.

A.9 Creating a Theme Step by Step

Here is the full process from start to a working theme:

1. Create a new folder inside `themes/` with a short, lowercase, hyphenated name — for example `themes/ocean-blue/`. The folder name becomes the theme's identifier in the database.
2. Copy `ensemblelogo.png` and `placeholder.png` from `themes/default/` into your new folder.
3. Create `style.css` in your new folder. Start with the import line and an empty `:root` block:

```
themes/ocean-blue/style.css — starting point
@import url('../default/style.css');  :root {     /* Your overrides
go here */ }
```

4. Open `themes/default/style.css` and find the `:root { }` block. Copy the tokens you want to change into your own `:root` block and edit their values.
5. Open the Ensemble web panel in a browser. Go to **Settings › Theme Selection**. Your theme appears in the list. Select it and click **Apply Theme** to preview it.
6. Adjust your CSS, reload the page, and repeat until you are satisfied.
7. (Optional) Take a small screenshot crop of the interface in your theme and save it as `preview.png` in your theme folder.
8. Your theme is complete.

Browser developer tools: Keep your browser's developer tools open while editing. The Elements panel lets you inspect any part of the UI to find class names and see which CSS properties are currently applied. The Styles panel shows which variable is resolving to which value, making it easy to track down a colour that is not changing as expected.

A.10 Sharing a Theme

A theme is entirely self-contained in its folder. To share it:

9. Zip up the theme folder, for example `ocean-blue.zip` containing the `ocean-blue/` directory.
10. Distribute the zip file — via the OpenSimWorld Script Library, a file hosting service, or directly to users.
11. Recipients extract the zip into their `themes/` directory. The theme appears in their Theme Selection modal immediately, with no code changes or server restart needed.

The recipient's directory structure after installation:

```
themes/
   default/          ← shipped with Ensemble
   dark/             ← shipped with Ensemble
   ocean-blue/       ← community theme, just extracted
```

Because theme discovery is automatic, no configuration file needs updating. The only requirement is that the folder is inside `themes/` and contains a `style.css`.

> **OpenSimWorld Script Library:** If you create a theme you are happy to share with the wider Ensemble community, consider uploading it to the OpenSimWorld Script Library alongside your description. Community themes benefit everyone using Ensemble and are a good way to contribute without needing to know PHP or LSL.

A.11 Complete Example: Ocean Blue Theme

The following is a complete, ready-to-use example theme with a blue ocean colour palette. It overrides both the palette tokens for full coverage and adds one structural override for the logo on the dark-ish surface.

themes/ocean-blue/style.css

```
@import url('../default/style.css');
:root {
     /* ── Palette — blue/teal replacing rose/periwinkle ── */
     --rose-100:  #f0f8ff;   /* alice blue */
     --rose-200:  #b8dff5;
     --rose-300:  #7cc0e8;
     --rose-400:  #3d9fd4;
     --rose-500:  #1a7eb8;   /* primary blue */
     --rose-600:  #105c8a;   /* darker on hover */

     --peri-100:  #eef6fb;
     --peri-200:  #b8d8ee;
     --peri-300:  #6daed8;
     --peri-400:  #3d86b8;
     --peri-500:  #1e6490;

     /* ── Semantic ──────────────────────────────── */
     --color-bg:              #f5f9fc;
     --color-surface:         #ffffff;
     --color-border:          var(--rose-200);
     --color-primary:         var(--rose-500);
     --color-primary-hov:     var(--rose-600);
     --color-accent:          var(--peri-400);

     /* ── Status ────────────────────────────────── */
     --color-online:  #0ea370;
     --color-stale:   #d4820a;
}
```

Save this as `style.css` in `themes/ocean-blue/`, copy `ensemblelogo.png` and `placeholder.png` from `themes/default/`, and the theme is ready to use. Every element of the interface — buttons, borders, outfit cards, the header bar, modal windows, status badges — will reflect the ocean blue palette.

This concludes the Ensemble Full Manual. Thank you for using Ensemble.