



MD Skin Studio

FULL USER GUIDE

Version 0.1 - Build 18

Apple Silicon



Native visual-profile editor for MD Record Studio

Application window

Contents

01	1. Product scope and purpose	4
02	2. Installer image and download verification	5
03	3. Installation and first launch	6
04	4. Main window	7
05	5. Preparing a safe workspace	8
06	6. Menus and keyboard commands	9
07	7. Settings and Runtime Log	10
08	8. New, Open, and Save	11
09	9. Duplicate, Undo, and Redo	12
10	10. Shared skin library	13
11	11. Built-in profiles and choosing a base	14
12	12. Live Preview and demonstration data	15
13	13. Canvas, zoom, rulers, and grids	16
14	14. Selecting elements and modules	17
15	15. Moving, resizing, and multiple selection	18
16	16. Element Inspector: Position and Size	19

Contents

MD Skin Studio

17	17. Element Inspector: text and typography	20
18	18. Element Inspector: appearance	21
19	19. Shadows and depth	22
20	20. Images and content mode	23
21	21. Module Inspector	24
22	22. Eight-module master map	25
23	23. Skin Inspector	26
24	24. `.mdskin` structure	27
25	25. Semantic IDs and overrides	28
26	26. Compatible JSON	29
27	27. Installation in MD Record Studio	30
28	28. Finished-skin QA checklist	31
29	29. Troubleshooting	32
30	30. Limits, support, and safe operation	33

Documented version: 0.1 build 18. Platform: Apple Silicon. Edition: 2026.

1. Product scope and purpose

MD Skin Studio is the public name of the native visual-profile editor for MD Record Studio. The current verified application still uses the technical name [MD Record Skin Studio](#), is installed at [/Applications/MD Record Skin Studio.app](#), and has bundle identifier [com.avd.MDRecordSkinStudio](#).

The application creates, opens, edits, and packages [.mdskin](#) documents. Its central workspace renders the real MD Record Studio SwiftUI interface in a safe demonstration state. The editor changes semantic controls - text, buttons, displays, indicators, images, fonts, borders, and shadows - without replacing functional UI with a static screenshot.

Preview does not control a physical deck. Device identity, disc capacity, and the eight tracks are demonstration data, and runtime-button hit testing is disabled. No MiniDisc or NetMD hardware is required to design a skin.

The supported workflow

- create a profile on the MD S-980 AD master layout;
- derive a variant from an existing profile;
- edit individual elements and module appearance;
- embed images and fonts;
- save a portable [.mdskin](#);
- export compatible JSON;
- install the profile in the MD Record Studio shared library;
- diagnose operations with the extended runtime log.

2. Installer image and download verification

The public installer is distributed as a DMG with a separate [.dmg.sha256](#) file. Treat these as one verification pair. Compare the downloaded DMG hash with the sidecar before installation.

Terminal verification

```
cd ~/Downloads  
shasum -a 256 -c "MD Skin Studio v0.1 build 18 Amber Layer Foundry.dmg.sha256"
```

The expected result is the DMG name followed by [OK](#). If verification fails, do not open the image. Remove the incomplete copy and download both files again from the same source.

DMG contents

- [MD Record Skin Studio.app](#) - technical application name;
- [Applications](#) - link to the Applications folder;
- [Manuals](#) - Russian and English guides;
- [README.txt](#) and [CHANGELOG.txt](#).

The DMG is locally ad-hoc signed and is not confirmed as Apple-notarized. This does not by itself mean the file is damaged, but macOS may require an additional first-launch confirmation. Use only an image whose hash matches the published sidecar.

3. Installation and first launch

1. Open the verified DMG.
2. Drag [MD Record Skin Studio.app](#) to [Applications](#).
3. Wait for copying to finish and eject the image.
4. Open the app from [/Applications](#), not directly from the DMG.
5. If macOS blocks the first launch, open System Settings ☐ Privacy & Security and confirm only after checking the SHA-256 and source.

The project deployment target is macOS 13.5. The verified build 18 artifact is [arm64](#) and has been checked on Apple Silicon under macOS Sequoia 15.7.7. Intel and Universal runtime are not confirmed.

At first launch, the app creates shared-library and runtime-log folders. It may also create or migrate bundled [.mdskin](#) packages. Back up any custom library before opening the editor.

Important paths

Application:

`/Applications/MD Record Skin Studio.app`

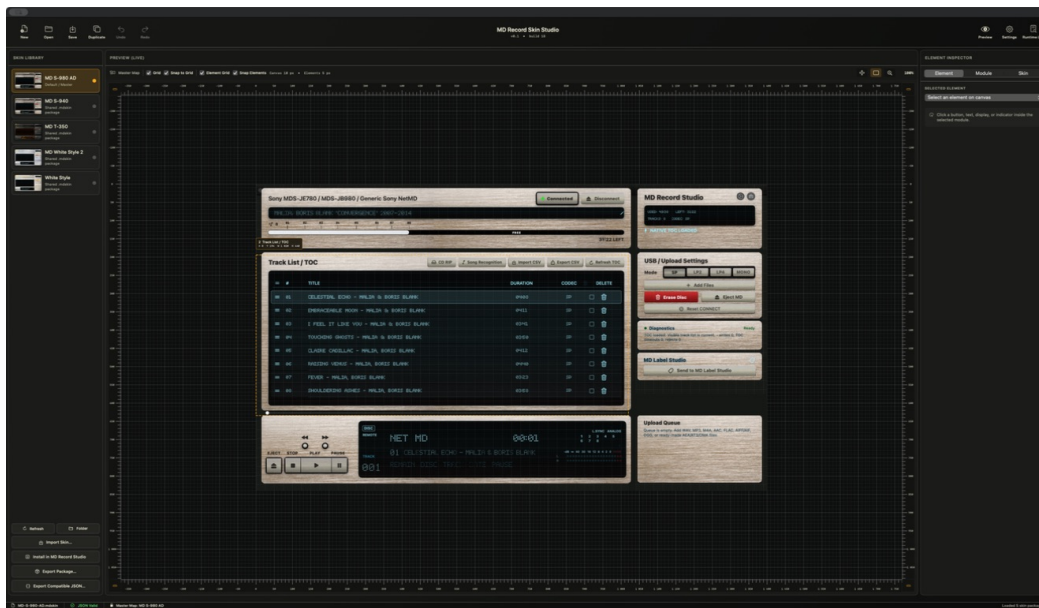
Shared library:

`~/Library/Application Support/MD Record Studio/Skins`

Runtime logs:

`~/Library/Application Support/MD Record Skin Studio/Logs`

4. Main window



Main window: library, live preview, canvas and inspector.

The main window contains four work areas and a bottom status bar.

1. Toolbar - New, Open, Save, Duplicate, Undo, Redo, Preview, Settings, and Runtime Log.
2. Skin Library - available profiles plus import and export commands.
3. Preview (Live) - the real MD Record Studio runtime, master map, grids, and editor overlays.
4. Inspector - Element, Module, and Skin tabs.
5. Status Bar - document name, JSON validity, master map, unsaved state, and last operation.

The window is designed for a large display but is clamped to the current monitor's visible frame. Use 100% zoom for final geometry checks.

Reading status

- **JSON Valid** - the document serializes under the current contract;
- **Validation Error** - do not treat the package as install-ready;
- **Master Map: MD S-980 AD** - the fixed eight-module map is active;
- **Unsaved changes** - current data differs from the last save;
- the rightmost message confirms the latest operation.

5. Preparing a safe workspace

Before editing a custom profile, copy its [.mdskin](#) to a separate folder. Never use the only original as the working document.

Suggested layout

```
Skin Project/  
└─ Original/  
└─ Working/  
└─ Exports/  
    └─ QA/
```

Keep an immutable source in [Original](#), edit a copy in [Working](#), save transferable revisions in [Exports](#), and store screenshots, comparison JSON, and QA notes in [QA](#).

Before broad changes, copy the complete [~/Library/Application Support/MD Record Studio/Skins](#) directory. Launching the editor can create, update, or migrate bundled packages.

Change one logical layer at a time: geometry, typography, colors, then assets. Save a distinct revision after each layer and confirm [JSON Valid](#).

6. Menus and keyboard commands

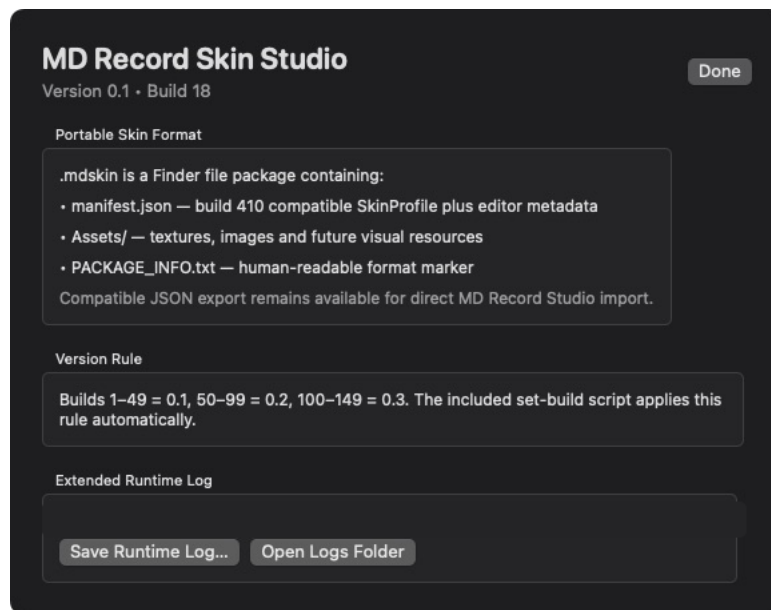
Core actions are available in both the toolbar and macOS menus.

Command	Shortcut	Purpose
New Skin	Command-N	Create a profile
Open Skin	Command-O	Open .mdskin or JSON
Save	Command-S	Save to current location
Save Skin Package As	Shift-Command-S	Save a new .mdskin
Undo	Command-Z	Revert a change
Redo	Shift-Command-Z	Restore a reverted change
Export Compatible JSON	Shift-Command-E	Export JSON

The Skin menu contains Duplicate Skin, Install in MD Record Studio, Refresh Shared Skins, Open Shared Skins Folder, Open Runtime Logs Folder, Reset Selected Element, Reset Selected Module, and Restore Master Layout.

Reset commands modify the document. Save a copy or confirm Undo is available first. Restore Master Layout returns canonical module geometry; it does not revert every text and color edit.

7. Settings and Runtime Log



Settings window: package format, version and runtime-log controls.

Settings opens as a sheet inside the application window. It shows the current version, portable [.mdskin](#) contents, build-version rule, and active runtime-log path.

Portable Skin Format

This section identifies [manifest.json](#), [Assets](#), and [PACKAGE_INFO.txt](#). Compatible JSON remains a separate transfer form without the package wrapper.

Extended Runtime Log

Each launch creates a timestamped text file containing version/build and the operation sequence. Save Runtime Log exports a copy through the system save panel. Open Logs Folder opens the log directory.

Before sending a log to support, review it and remove personal paths, user-file names, or other private data. The Settings screenshot in this guide already redacts the personal home-path prefix.

A runtime log does not replace a [.mdskin](#) backup. It explains operations but cannot reconstruct a package automatically.

8. New, Open, and Save

New

New creates [Untitled Skin](#) on the fixed MD S-980 AD master layout. It has no save location. Set a meaningful name, author, and notes in Skin Inspector.

Open

Open accepts a [.mdskin](#) package or compatible JSON. The format is validated before replacing the current document. If [formatVersion](#) is newer than supported, a safe warning appears and the current document remains loaded.

Save

Save writes to the current location. A new profile opens a [.mdskin](#) save panel. Save Skin Package As always selects a new destination and should not replace the only source without a backup.

After saving, confirm:

- filename and profile name agree;
- [manifest.json](#) exists;
- Status Bar reports [JSON Valid](#);
- imported files are inside [Assets](#);
- the package opens again.

9. Duplicate, Undo, and Redo

Duplicate creates a copy with a new identifier and [Copy](#) suffix. It does not reuse the source save location and must be saved as a new package.

Undo and Redo use complete document snapshots. History retains up to 100 states, and adjacent changes may be coalesced. This is useful for comparing visual alternatives but is not a substitute for disk revisions.

Comparison technique

1. Record the original value.
2. Change one field.
3. Inspect Preview at 100%.
4. Toggle Undo and Redo several times.
5. If accepted, save a new package revision.

A build-18 observation remains open: [Unsaved changes](#) can persist after a full Undo in some scenarios. If preserving the source is critical, do not press Save; close the document and compare the working package with its backup.

10. Shared skin library

Both applications use:

```
~/Library/Application Support/MD Record Studio/Skins
```

The editor enumerates immediate `.mdskin` child directories and decodes their `manifest.json`.

Library commands

- Refresh - scan the folder again;
- Folder - open the skins directory;
- Import Skin - validate and copy a package or JSON into the shared library;
- Install in MD Record Studio - write the current document into the library;
- Export Package - save a portable copy;
- Export Compatible JSON - save the profile without the package wrapper.

Packages with the same stable `id` may compete as one logical profile. Use Duplicate and verify a new ID for an independent variant.

11. Built-in profiles and choosing a base



MD T-350 selected in the library with a valid package state.

The verified library contains:

- MD S-980 AD - default/master;
- MD S-940;
- MD T-350;
- MD White Style 2;
- White Style.

Selecting a card loads its document, clears selection and Undo/Redo history, then refreshes Preview. Save unfinished work first.

Choosing a base

MD S-980 AD is useful as a neutral starting point and master-map reference. MD T-350, MD S-940, and the light profiles provide examples of typography, contrast, and materials.

Do not change [visualPreset](#) manually to swap asset sets. Skin Inspector explains that the visual identifier and canonical assets come from the imported package and synthetic preset switching is disabled.

The amber dot identifies the active profile. After selection, verify the filename, [JSON Valid](#), and library-count message.

12. Live Preview and demonstration data

Preview renders the functional MD Record Studio SwiftUI runtime, not a module screenshot. The editor adds selection areas, frames, and resize handles.

The built-in model runs with [isSkinStudioPreview](#). Runtime buttons ignore input. NetMD connection, TOC reading, recording, and disc erasure do not occur.

Appropriate Preview checks

- geometry and alignment;
- typography and readability;
- colors, material, border, opacity, and shadows;
- image content mode;
- visibility and layer;
- master-map compliance.

Not proven by Preview

- physical deck communication;
- real disc time and tracks;
- USB/NetMD transfer;
- actual command-button behavior;
- MD Record Studio performance on another Mac.

13. Canvas, zoom, rulers, and grids

The logical canvas is 1400 × 840. Modules follow a 10 px canvas grid. Elements use an independent 5 px grid.

Available zoom levels are 82%, 100%, and 118%. Toolbar Preview toggles 100% and 118%. Use 100% for measurement and final approval; 82% shows the whole interface and 118% helps inspect details.

Toggles

- Grid - show the canvas grid;
- Snap to Grid - canvas snapping;
- Element Grid - show the module-level grid;
- Snap Elements - snap element frames.

Four rulers remain fixed to the viewport edges and measure from the centered skin origin. Editor grids do not appear in the exported runtime.

14. Selecting elements and modules

Click a visible runtime region to select a semantic element. Hit testing uses the effective element and module frames.

Confirmed types:

- text;
- button;
- display;
- image;
- indicator.

When elements overlap, hit testing considers size and logical rank. The selected control maps to a stable semantic ID such as [transport.playButton](#) or [recordStudio.statusDisplay](#).

Switch to Module Inspector or click module space outside an element to select a module. Moving from Element to Module clears the current element selection.

If selection is difficult, return to 100%, enable Element Grid, and use the [Selected Element](#) picker.

15. Moving, resizing, and multiple selection

Elements can move inside their parent module. Eight handles resize sides and corners. Minimum size is 5 px.

Frames are clamped to module bounds. Invalid input is normalized so an element cannot accidentally leave its module.

Selection and multi-selection are supported. Inspector reports the selected count, and reset can apply to the selected group.

Precise sequence

1. Select an element and record X, Y, Width, and Height.
2. Enable Snap Elements.
3. Move or resize.
4. Verify numeric fields.
5. Disable snapping only for final micro-adjustment.
6. Compare with Undo/Redo.

16. Element Inspector: Position and Size



Element Inspector: frame, text, appearance, assets and shadow.

The top of Element Inspector shows name, type, and semantic ID. Position & Size provides X, Y, Width, and Height fields and sliders.

Coordinates are logical pixels relative to the parent module's top-left corner, not absolute screen coordinates.

Constraints

- minimum width and height are 5 px;
- frame remains inside module bounds;
- Snap Elements rounds to the element grid;
- module geometry is separate and fixed;
- changing layer does not change frame.

Set Width and Height first, then X and Y. Re-check width after changing text, font, or font size. Reset Selected Element restores source-profile frame and appearance.

17. Element Inspector: text and typography

Text elements expose value, font family, font size, weight, alignment, and text color.

Font family

The list includes system, bundled, and imported families. If a package names an unavailable family, the editor retains the name but Preview may use a system fallback.

Weight and alignment

Supported weights include regular, medium, semibold, and bold. Alignment is leading, center, or trailing.

Recommendations

- test digits, Cyrillic, and Latin text;
- avoid thin text on textured material;
- preserve internal button and display padding;
- test long device and track names;
- review all zooms, but approve geometry at 100%.

Check redistribution rights before importing a custom font because the font file becomes part of the [.mdskin](#).

18. Element Inspector: appearance

Appearance combines material, material tint, background, border, corner radius, opacity, layer, visibility, and image mode.

Material

[Original](#) / [Inherit](#) preserves the source material. Model cases include brushed metal, dark metal, light metal, glass, and solid. Exact runtime application depends on MD Record Studio.

Border and corners

Border color and width define the outline. Corner radius rounds the element; excessive values can turn a short button into a capsule.

Opacity, layer, visibility

Opacity ranges from transparent to opaque. Layer controls overlap order. Visibility hides the element while retaining configuration.

Material Tint uses the system ColorPicker. Automated entry is limited by macOS ColorPicker accessibility; review the visual result and save before broad tint changes.

19. Shadows and depth

Element shadow fields include color, opacity, blur radius, and X/Y offset.

Shadows should separate a control from its material without obscuring text. Start with low opacity and blur, then change one parameter at a time.

Common mistakes

- a large shadow blurs adjacent controls;
- negative offset suggests an inconsistent light source;
- a dark shadow disappears on a black display;
- glow around small text harms readability;
- another element's layer can cover the shadow.

Review the shadow wherever the element crosses different background tones. Use consistent values for controls in the same group.

20. Images and content mode

Choose Image opens a picker and copies the file into the current package's [Assets/Images](#). The manifest path must remain relative.

Content modes

- fit - show the complete image, possibly with empty space;
- fill - fill the frame, possibly cropping edges;
- stretch - fill width and height, possibly changing proportions.

Clear removes a custom binding and returns to the source resource or empty state.

After import:

1. save the package;
2. inspect [Assets/Images](#);
3. confirm there is no absolute path;
4. close and reopen;
5. inspect Preview;
6. test the installed copy in MD Record Studio.

Whole-module screenshots are not live-renderer content and must not replace a functional module.

21. Module Inspector



Master Map and Module Inspector with fixed module bounds.

Module Inspector edits selected-module appearance:

- material;
- background and accent;
- border color and width;
- corner radius;
- opacity;
- layer;
- visibility.

Position & Size are read-only because [md-s980ad-master](#) owns module geometry.

Reset Module restores the current module's appearance. Restore Entire Master Layout restores the canonical eight-module map.

Design major modules first, then their elements. Changing module material, opacity, and element colors together makes contrast problems difficult to diagnose.

22. Eight-module master map

Module	X	Y	Width	Height
Header	0	0	1020	180
Track List / TOC	0	194	1020	440
Transport	0	648	1020	192
MD Record Studio	1040	0	360	180
USB / Upload Settings	1040	194	360	178
Diagnostics	1040	386	360	88
MD Label Studio	1040	488	360	86
Upload Queue	1040	648	360	192

Stable IDs are [header](#), [trackList](#), [transport](#), [recordStudio](#), [uploadSettings](#), [diagnostics](#), [labelStudio](#), and [uploadQueue](#).

The master map keeps functional MD Record Studio regions in expected locations. Format 1 does not support arbitrary module rearrangement. Use module appearance and element frames for composition.

23. Skin Inspector



Skin Inspector: identity, author, notes and format contract.

Skin Inspector controls document identity:

- name;
- author;
- notes;
- visual preset ID;
- format/layout/editor contract summary;
- Export Compatible JSON.

Use a descriptive name distinct from the source profile. Author and Notes aid maintenance; never include passwords, email addresses, serial numbers, or private paths.

Visual preset ID is tied to the imported package and canonical assets. Synthetic preset switching is disabled, so do not use this field as an arbitrary theme switch.

Known build-18 discrepancy: the UI displays [Editor schema 2](#), while the current model and serialization use schema 3. This is a stale UI label, not support for two current schemas.

24. .mdskin structure

```
Skin Name.mdskin/  
└─ manifest.json  
└─ Assets/  
    └─ Images/  
        └─ Fonts/  
            └─ PACKAGE_INFO.txt
```

[manifest.json](#) is required. [Assets](#) stores portable images and fonts. [PACKAGE_INFO.txt](#) identifies the package for people and recovery tools.

Current [formatVersion](#) is 1 and layout is [md-s980ad-master](#). Editor metadata uses schema 3 and stores canvas, grid, semantic elements, element overrides, module overrides, embedded fonts, and relative asset paths.

Readers must ignore unknown optional fields. Writers must keep root profile fields valid. An incompatible change requires a new [formatVersion](#).

25. Semantic IDs and overrides

Each runtime control has a stable ID such as `header.deviceTitle`, `trackList.durationHeader`, `transport.playButton`, or `recordStudio.usedText`.

`elements` describes the complete program-provided semantic map. `elementOverrides` stores user changes. When an ID is absent from overrides, runtime keeps the value from `elements`.

Module overrides work similarly, while geometry remains under the master map. Do not rename stable IDs manually because that breaks the link to functional SwiftUI controls.

If you inspect JSON manually, work on a copy and validate syntax. Avoid global ID search-and-replace without an open-and-save test.

26. Compatible JSON

Export Compatible JSON writes the profile without the package wrapper. This is useful for diagnostics, comparison, and direct integration, but JSON alone cannot carry external assets unless their expected relative files are also available.

Root fields include [formatVersion](#), [id](#), [name](#), [visualPreset](#), [layoutPreset](#), [author](#), [notes](#), and optional [editor](#).

Use JSON for

- revision diff;
- developer handoff without Finder packaging;
- schema and semantic-ID inspection;
- MD Record Studio bridge work.

Use .mdskin for

- user distribution;
- portable images and fonts;
- shared-library installation;
- archiving a finished profile.

After JSON export, open a copy in the editor and confirm [JSON Valid](#).

27. Installation in MD Record Studio

1. Save the working [.mdskin](#).
2. Back up the shared library.
3. Confirm [JSON Valid](#).
4. Click Install in MD Record Studio.
5. Open or refresh the MD Record Studio skin list.
6. Select the installed profile.
7. Compare appearance with Preview.
8. Test functional buttons in MD Record Studio.

If the result is wrong, restore the backup package and refresh the library. Do not replace an original bundled package with the only custom copy.

Preview verifies appearance, but final round-trip requires real MD Record Studio. Physical NetMD hardware is needed only for Record Studio functions, not skin packaging.

28. Finished-skin QA checklist

Document

- unique name and ID;
- no private data in author or notes;
- `formatVersion = 1`;
- `md-s980ad-master` layout;
- `JSON Valid`;
- separate backup.

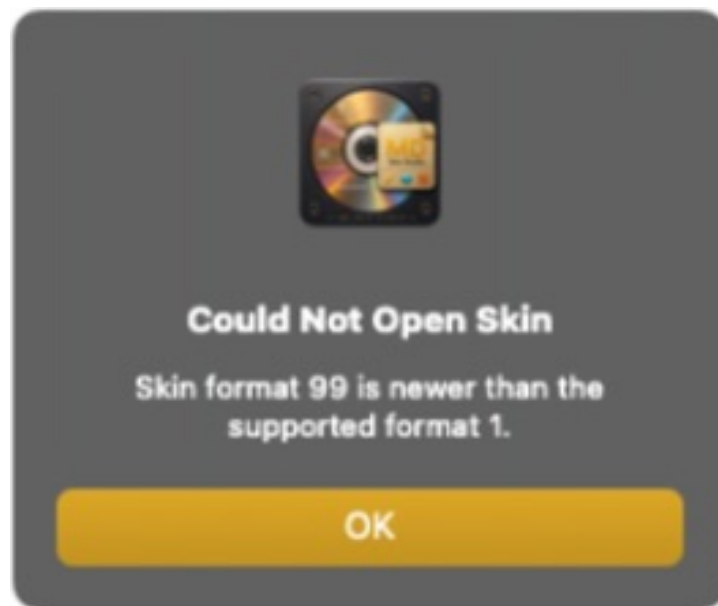
Visual

- text is readable on every material;
- long strings are not clipped;
- button sizes are consistent;
- displays and indicators do not overlap;
- visibility and layer are intentional;
- image proportions are correct;
- shadows preserve readability.

Portability

- images are in `Assets/Images`;
- fonts are in `Assets/Fonts`;
- paths are relative;
- package reopens;
- MD Record Studio sees the installed copy;
- compatible JSON exports cleanly.

29. Troubleshooting



Safe refusal of a package that uses an unsupported format.

Skin format N is newer than supported format 1

The file was created by a newer format. Do not lower the number manually or overwrite it with build 18. Use a compatible editor version.

Package is missing from the library

Click Refresh, open Folder, check the [.mdskin](#) extension and [manifest.json](#), then inspect the runtime log.

Element cannot be selected

Return to 100%, open Element Inspector, select the correct module, and use the Selected Element picker.

A change is not visible

Compare Undo/Redo, confirm you changed an element rather than a module, and check layer/visibility. Do not save an uncertain version over the original.

Module frame will not change

Expected: format 1 uses a fixed master map.

Image or font is missing

Inspect Assets and relative paths, then reopen the package. Use the built-in import commands rather than linking an external file.

30. Limits, support, and safe operation

The current verified artifact is build 18 arm64. A fresh Release from the current source graph is blocked by external MD Record Studio audio-meter source drift. The current master is a verified Sequoia-specific resource refresh of the previous build 18, not a newly compiled Release.

Intel/Universal runtime, Developer ID signing, and notarization are not confirmed. The full WYSIWYG acceptance matrix is incomplete. Exhaustive real custom-image/font round-trip and automated Material Tint input are not fully closed.

Include with a support request

- public name MD Skin Studio;
- technical version/build;
- macOS version and Mac model;
- exact operation sequence;
- runtime log with private paths removed;
- a transferable test `.mdskin`, if possible;
- screenshot of only the application window or specific warning.

Never send a password, signing key, token, private disk, personal path, or the only original package. Keep working and backup copies separate.