

ADMINISTRATOR / OPERATOR EDITION

myOrganizer: CURATOR Admin Guide

Configuration, operations, evidence, validation, and recovery

Collection Utility for Retro Archive Tracking, Organization, Repository

Updated: October 11, 2026 (v2.5.0-beta.1)

Created, architected, and managed by Miguel A. Ortiz

Development and coding assistance: OpenAI ChatGPT

Third-party tools, authority data, and trademarks remain credited under their respective notices.

Copyright © 2026 Miguel A. Ortiz. All rights reserved except as granted by the applicable license.

myOrganizer: CURATOR Safety Notice — Read Before Use

CURATOR reorganizes, stages, extracts, packages, and writes collection files.

- **Do not use original archive or game files as the Source/Input folder. Admins should require operators to run CURATOR against copies.**
- **Only replace original files after output review, validation, and emulator confirmation.**
- **Before large runs, confirm Source, Collection, and Workspace capacity. Archive extraction and disc-media work can require substantial temporary space.**

Public Release Notice

CURATOR is released under the GNU General Public License, version 3 or later, and is provided without warranty. Administrators are responsible for protecting files, preserving evidence, validating output, and maintaining third-party license/provenance records.

Modified versions must be clearly marked as modified and must not imply that they are official CURATOR releases by Miguel A. Ortiz unless explicitly authorized.

Copyright © 2026 Miguel A. Ortiz

Project Links

Use the official CURATOR website for product information and release updates. Use the GitHub page for public release announcements, licensing information, documentation, and source availability as the project is published.

- Official website: <https://myorganizerhq.com/>
- GitHub page: <https://github.com/Jiraiya1969/CURATOR>

Contents

Section	Topic
1	Purpose and audience
2	Administration model
3	Gateway control surface
4	Configuration and operator overrides
5	Folders and data ownership
6	System requirements and dependencies
7	Authority baseline operations
8	Run operations
9	Stage 5 recovery administration
10	Maintenance procedures
11	Evidence, logs, and validation
12	Stage 10 and output administration
13	Security and data safety
14	Troubleshooting reference
15	Admin checklists
Current	Current collection behavior
Appendix A	Tool and dependency references
Appendix B	Credits, sources, and third-party attributions

Critical Safety Warnings

These warnings apply throughout this guide. Read them before configuring folders or starting a run.

SOURCE PROTECTION - Require operators to use copies, never irreplaceable originals, as Source/Input. Keep verified backups, use one primary source location per run, and review output before replacing or relocating source material.

OUTPUT REPLACEMENT - Clear is a replacement decision: it removes existing contents from the configured collection destination before publication. Use Clear only when that destination is safe to replace. Use Append when existing output must remain; it uses discovery and retained database receipts for known unchanged targets. Keep output unchanged during the run; same-size external changes are not detected by ordinary Append.

PRESERVE INCOMPLETE SETS - Keep this folder and its contents in the collection output. Manually removing retained incomplete-set archives can prevent a later Append or Resume run from completing.

EVIDENCE AND CLEANUP - Do not move, rename, edit, or delete relational authority, active run state, workspace, logs, placement receipts, manifests, or reports while CURATOR is operating. Archive or cleanup actions can invalidate recovery and audit continuity; run them only while stages are idle and only after confirming the intended evidence loss.

AUTHORITY, RUNTIME, AND REVIEW - Deterministic authority matching can take hours, with runtime scaling according to the volume and size of the files being processed. MAME support in myOrganizer: CURATOR is a VERY EARLY, experimental implementation. It must NOT be relied upon as a proven means of organizing MAME sets. Continue using clrmame as the definitive MAME ROM manager. Stage 5 attempts supported CDI, NRG, and MDF/MDS decoding through the bundled managed Aaru libraries; any decoder-incompatible image is logged, retained unchanged, and continues through the standard matching and review path. Never force identity from a filename or conversion, and review warnings and final outputs before release.

CDI DAT MATCHING REMAINS UNVERIFIED - No tested Dreamcast or Jaguar CD CDI has matched a loaded DAT, either as the original image or through its decoded tracks. Successful decoding or conversion does not confirm DAT identity or playback compatibility. Review unmatched CDI files and their generated outputs.

1. Purpose and Audience

This Admin Guide is for the person responsible for operating, configuring, validating, maintaining, and release-checking CURATOR. It assumes the administrator understands the project folder, PowerShell execution, authority data ownership, generated JSON evidence, workspace data, destination policy, dependency provenance, and final physical output.

This guide covers myOrganizer: CURATOR v2.5.0-beta.1.

- Configuration schema: 1.0.30

Admin responsibility: protect deterministic behavior. Do not change configuration, operator overrides, DAT files, platform files, tools, dependencies, run-state, or generated JSON without understanding which downstream stages depend on that evidence.

2. Administration Model

CURATOR is a staged local pipeline. Administration work is mostly about keeping authority inputs, runtime configuration, generated evidence, recovery state, and output locations aligned. The gateway is the preferred control surface for normal operators and most admin actions.

Administrative area	Admin owns
Authority data	DAT files, platform XML/reference files, and decisions about when Stages 1-3 must be rebuilt.
Path policy	Source, Collection, and Workspace choices through gateway-managed operator overrides.
Configuration contract	config.json schema, stage order, dependency paths, tool paths, and core policy settings.
Runtime evidence	Relational stage state, logs, history archives, run_state records, final reports.
Workspace hygiene	When workspace, _temp and stage History contents may be cleared.
Output safety	Destination policy, Clear vs Append, NeedsAttention routing, physical output failures, and final report preservation.

3. Gateway Control Surface

Start the gateway from the project root with `pwsh -File .\Curator.ps1`. The startup safety notice is mandatory; the operator must type `AGREED` before the menu appears.

On a clean installation, `database\CURATOR.db` does not yet exist. The Gateway shows only `P` — Prepare DAR. Other menu choices are hidden and their keys are not accepted until the database exists.

Choose `P` and confirm the Prepare DAR run. Prepare DAR does not request Clear or Append and does not prepare collection output. It runs Stages 1–3: Stage 1 creates and initializes the database and prepares platform references, Stage 2 discovers DAT files, and Stage 3 publishes the validated DAR authority.

The full menu becomes available when the Gateway refreshes and finds the database. Its appearance does not prove that all three preparation stages succeeded. If Prepare DAR fails, correct the reported problem and run `P` again; choose `O` — Organize only after Prepare DAR completes successfully. Stage 2 hashes every original DAT/XML and ingests only new files or files whose content hash changed. Application or mapping changes alone do not re-ingest unchanged files. Stages 1 and 3 retain their own source and policy checks.

RunIDs identify execution and provenance; a new RunID alone does not invalidate retained authority.

The menu reference below describes the full menu after database creation. Check setup, Manage folders, reporting, maintenance, and individual-stage options are unavailable from the first-run menu.

Choice	Admin meaning	Use it when
P Prepare DAR	Run Stages 1-3 to refresh platform and DAT authority.	Use when the described operation is required.
O Organize	Run Stages 4-10 against the committed relational authority revision.	Use when the described operation is required.
1 Full run	Run all configured stages without intermediate stage pauses. Final-stage and Gateway summary acknowledgements remain.	Use when the described operation is required.
2 Full run with pause	Run all configured stages. Press Enter before and after each stage summary, including the final stage; acknowledge the Gateway run summary separately.	Use when the described operation is required.
3 Run one stage	Run one selected stage.	Use when the described operation is required.
4 Run one stage with pause	Run one selected stage. Press Enter before and after its stage summary; acknowledge the Gateway run summary separately.	Use when the described operation is required.
5 Run from stage	Run from a selected stage through the end.	Use when the described operation is required.
6 Run from stage with pause	Run from a selected stage through the end. Press Enter before and after each stage summary, including the final stage.	Use when the described operation is required.
7 Run through stage	Run from Stage 1 through a selected stage.	Use when the described operation is required.
8 Run through stage with pause	Run from Stage 1 through a selected stage. Press Enter before and after each stage summary, including the final selected stage.	Use when the described operation is required.
9 Check setup	Check project paths, configuration, dependencies, and stage readiness.	Use when the described operation is required.
10 Current status	Show the current configured folder and latest run status.	Use when the described operation is required.
R Collection coverage report	Open current Reporting-generated company coverage results.	Use when the described operation is required.
V Full physical collection audit	Hash every collection artifact and review physical differences before any manifest update.	Use after manual collection changes or when content-level verification is required.
H Clear histories/temp/workspace	Run the confirmed maintenance cleanup workflow.	Use when the described operation is required.
D Generate API document	Regenerate the Core API Developer Handbook source document.	Use when the described operation is required.
S Clear run states	Clear supported saved run-state records after confirmation.	Use when the described operation is required.
F Manage folders	Change Source, Collection, and Workspace operator paths.	Use when the described operation is required.
X Advanced options	Open advanced maintenance and diagnostic options.	Use when the described operation is required.
Q Exit	Close the gateway.	Use when the described operation is required.

Gateway/Core lifecycle continuity is preserved across supported child launches. Returning from a separately launched stage does not require closing PowerShell; full-run setup remains available subject to its ordinary readiness and confirmation gates.

Start actions from the Gateway. If startup reports a missing or incompatible dependency, restore the complete application package before retrying.

4. Configuration and Operator Overrides

ADMIN WARNING - CONFIGURATION CONTRACT

config\config.json is the source contract for schema, stage order, dependency paths, tool paths, JSON behavior, platform media category mapping, matching policy, optical/container behavior, and Organizer behavior. It is not the normal UI edit surface for operator folders.

config\operator_overrides.json is the gateway-managed operator path surface. The allowed overrides are EnvironmentDefaults.Input, EnvironmentDefaults.Output, and Paths.WorkspaceFolder. When the override file exists and is valid, it is authoritative for all three paths as one complete contract. Missing or empty values fail validation and never fall back to config.json. All three configured folders must be outside the CURATOR application folder; the application folder is reserved for program files and must not contain volatile source, workspace, or collection data.

Proprietary optical-image handling is configured in config\config.json under FileHandling.Containers.OpticalDecode. Keep Enabled set to true to attempt decoding. Extensions defines the accepted .cdi, .nrg, .mds, and .mdf inputs; KeepSource must remain true; OnFailure must remain WarnAndContinue; and MemberMode must remain RawLongSectorTracks. These values preserve source custody and deterministic member bytes.

The optical decoder is embedded as six managed Aaru libraries under dependencies. Core.Database uses the pinned Microsoft.Data.Sqlite, SQLitePCLRaw, and native e_sqlite3 libraries from the same required dependency folder. No separate database installer, service, PATH entry, or user profile is required. If any required library is missing or fails integrity validation, restore the complete CURATOR dependencies folder rather than downloading a replacement into the live application.

Path	Admin handling
Source/Input	Must exist. Use copies of original collections.
Destination/Collection	May be created or reused. Apply Clear/Append policy deliberately.
Workspace	Temporary staging area. Must be distinct from Source and Collection.

Path	Admin handling
Distinctness	Source, Collection, and Workspace must be three distinct physical folders, all outside the CURATOR application folder.

1. Record the config version and reason for any contract-level change.
2. After first-run preparation, use Manage folders for normal Source/Destination/Workspace changes and Check setup before organizing. Use Prepare DAR again when platform or DAT authority inputs change. Generate API document is also available from the full menu.
3. Run Check setup after changing paths or required tools.
4. Run the smallest stage range that proves the change.
5. Inspect JSON summaries and stage logs before accepting the change.

5. Folders and Data Ownership

Folder	Role	Handling guidance
config	Configuration, templates, and operator overrides.	Treat config.json as controlled; operator_overrides.json is gateway-managed.
core	Core runtime modules.	Do not edit during normal administration.
scripts	Gateway and stage scripts.	Start stages 1–10 through the Gateway; direct stage launch is rejected.
DATs	DAT authority inputs.	User-supplied DAT authority payloads are not official release content. The only bundled exceptions are the eight individually allowlisted CURATOR-created authority DATs: the Atari 8-bit Preservation DAT, six Atarimania GameBase DATs for Atari 2600, 5200, 7800, Jaguar, Lynx, and ST, and the Atari Lynx Dragnet supplemental DAT. Obtain all other DATs from the appropriate source authority, organize deliberately, and rerun Stages 1-3.
platforms	Platform reference inputs.	User-supplied LaunchBox-style platform XML/reference payloads. Not official release content. Update deliberately and rerun Stages 1-3.
run_state	Gateway launch/status evidence.	Preserve during operation. Active recovery is read from Core.Database, not an ACTIVE_RECOVERY.json file.
workspace	Prepared/staged files.	May be wiped by Stage 5 clean start; preserved during Stage 5 resume.
_temp	Temporary runtime files.	Disposable only when no stage is active.
stage History	Archived stage logs/evidence.	Cleared by H only after confirmation.
Destination	Physical output and final report.	Protect with Append unless replacement is intended.

Folder	Role	Handling guidance
database	Persistent CURATOR.db authority and relational stage state.	Do not edit it manually.

6. System Requirements and Dependencies

CURATOR requires PowerShell 7.6.1 at minimum. PowerShell 7.6.1 and newer are allowed. Required runtime dependencies must remain under configured folders unless config is deliberately changed and revalidated. Missing or unstartable required tools are hard stops.

MDF/MDS LIMITATION - Alcohol MDF/MDS images vary by producer and descriptor/data pairing. CURATOR attempts the paired image when both members are available, but some otherwise valid pairs are incompatible with the bundled decoder and may report that the data file cannot be opened.

When MDF/MDS decoding fails, CURATOR records one warning, keeps both original members intact, and continues normal processing. The files may remain unmatched and be routed for review. This is expected warn-and-continue container behavior, not a stage failure and not proof that the source is corrupt.

Component	Purpose	Admin check
PowerShell 7.6.1 minimum; PowerShell 7.6.1 or later allowed	Gateway and stage runtime.	Confirm pwsh before operation.
AlphaFS	File-system services and long-path support.	Keep dependencies\AlphaFS.dll present.
HashLib4CSharp	Optional SHA-256 hashing services.	Keep dependencies\HashLib4CSharp.dll present.
Newtonsoft.Json / Schema	JSON read/write and validation support.	Keep both DLLs present.
Spectre.Console	Interactive console/progress rendering.	Use interactive PowerShell for visible validation.
7-Zip / 7z.dll	Persistent ZIP creation, archive listing, selective mapped extraction, and Core.Archive-owned CRC32/SHA-1 streaming.	Keep tools\7z.dll present. The retained 7z.exe is not required at runtime.
CHDMan	CHD extraction/creation.	Keep tools\chdman.exe present.
DolphinTool	RVZ effective-disc identity evidence.	Keep tools\DolphinTool.exe present.
BinMerge	Optional BIN/CUE helper.	Required only when that optional helper is used.

Component	Purpose	Admin check
Aaru managed image decoders	Reads supported CDI, NRG, and MDF/MDS optical images into deterministic raw track members.	Keep all six Aaru DLLs in dependencies; do not install or configure the Aaru CLI.

7. Authority Baseline Operations

DAT discovery reports completed DATs and the sets and members read for the current platform. Reference preparation reports its reuse check and summary loading. Checking unchanged sources may still require reading their contents.

Stages 1-3 create the authority baseline used by later curation runs. A stable baseline means the platform files and DAT files cover the collection universe the administrator intends to curate.

Authority inputs are split. User-supplied DAT files and large LaunchBox-style platform XML/reference payloads are local authority inputs, not permanent project source or official CURATOR release content. The eight individually allowlisted CURATOR-created authority files are required release files. CURATOR-supplied routing platform stubs under platforms are also official release support files when built-in output routing rules depend on them. Keep provenance clear, preserve the bundled CURATOR authority definitions, store other user-supplied authority data under the configured DATs and platforms folders, and rebuild authority whenever the intended authority set changes.

DAT Authority Determines Final Output

Output identification depends on the DAT files loaded during Stages 1–3. CURATOR retains this reference library in database\CURATOR.db. Files without an exact size and hash match are sent for review; CURATOR does not identify them by filename alone.

Administrators should expect real-world collections to contain variants that are not canonically documented in the selected DAT set: encrypted vs. decrypted dumps, trimmed or altered dumps, scene releases, hacks, homebrew, download-play files, bad dumps, and live database variants from different authorities. A known title can still require NeedsAttention handling when its exact content hash is absent from No-Intro, TOSEC, Libretro RDB exports, or any other loaded authority.

When DATs change, run Stages 1-3 before treating downstream authority as current. Stage 2 hashes eligible source contents before deciding which sources need parsing; only new files or files with changed original content hashes are ingested. Application, configuration and mapping changes alone do not select unchanged files for ingestion. Metadata-only timestamp changes do not invalidate content by themselves. This is not a constant-time change check: unchanged sources can still require full reads, and Stage 3 performs database comparisons before reuse.

WARNING: VERY EARLY MAME Implementation

MAME support in myOrganizer: CURATOR is a VERY EARLY, experimental implementation. It must NOT be relied upon as a proven means of organizing MAME sets. Continue using clrmame as the definitive MAME ROM manager.

MAME output uses SPLIT sets. Use clrmame to manage MAME collections:
<https://mamedev.emulab.it/clrmamepro/>

CURATOR accepts content-validated MAME -listxml exports for coin-operated Arcade ROM and CHD authority. It identifies the export from the MAME XML structure and metadata, not from an mame*.xml filename.

- Current output contract: split sets. Each machine retains its own ROM members while parent, BIOS, and device dependencies remain separate sets.
- CHD precedence: MAME internal CHD SHA-1 authority takes precedence over Redump-derived optical-track evidence on the MAME path.
- Fail-closed handling: nodump entries are excluded, and missing required members or companion archives retain the IncompleteAuthoritySet review reason. Their physical destination is under the root Incomplete Sets folder.
- Excluded scope: complete merged, non-merged, standalone, and general non-Arcade MAME dependency management. Supported software-list XML contributes exact member authority but does not make CURATOR a MAME-native set manager.

Use clrmame to organize, audit, and rebuild MAME sets. Do not rely on myOrganizer: CURATOR to establish that a MAME collection is complete or playable.

Provisioning Official MAME Software Lists

MAME support in myOrganizer: CURATOR is a VERY EARLY, experimental implementation. It must NOT be relied upon as a proven means of organizing MAME sets. Continue using clrmame as the definitive MAME ROM manager.

Provision the official per-list XML authority files. For Atari 2600, use:

-  a2600.xml
 -  a2600_cass.xml
1. Download the intended official MAME release from <https://www.mamedev.org/release.html> and extract it. Use one release consistently; do not mix software-list files from different MAME releases.
 2. Open the extracted MAME hash folder and locate the official per-list XML files.
 3. Copy  a2600.xml and  a2600_cass.xml into the appropriate authority folder, for example  DATs\Atari\MAME\Atari 2600. CURATOR reads the per-list XML without loading an external DTD.
 4. Run Prepare DAR, or, after database initialization, run Stages 1, 2, and 3 separately from the full menu. Review Stage 2 and Stage 3 evidence to confirm both software lists were accepted and published to Core.Database.

5. When updating MAME, replace both XML files from the same new release and rebuild Stages 1-3. Retain the MAME release number as provenance for the authority baseline.

Use the official per-list XML files from the MAME hash folder for these inputs. A separate DTD file is not required.

Acceptance checks

- Both XML files came from the same official MAME release and remain in one folder.
- Stage 2 identifies both XML files as MAME software lists without requiring an external DTD.
- Stage 3 publishes the selected list authority to Core.Database and links it to the current Stage 2 run.
- The authority baseline is rebuilt whenever either XML file changes.

Official references: <https://docs.mamedev.org/contributing/softlist.html>,
<https://docs.mamedev.org/commandline/commandline-all.html>, and
<https://www.mamedev.org/release.html>.

Provisioning Atari 8-Bit Preservation Authority

Unsupported HyperList inputs

Do not use the Atari 2600 HyperList XML or its derived DAT as authoritative ROM definitions, including copies from older packages.

The source XML does not declare complete member identities. Values inferred from other DATs are not declarations from HyperList.

Removing DAT input files does not remove previously stored authority or existing collection output.

CURATOR recognizes Atari 8-bit Preservation exports by their contents:

-  Atari 8-bit Software Preservation Initiative export - physical-release provenance with machine-readable dump hashes from <https://www.a8preservation.com/v2>
1. Generate or download the current machine-readable export from the authority's official site. Keep the unmodified export and record its retrieval date and source URL.
 2. Place each export anywhere under CURATOR's  DATs folder. Use a clear Atari 8-Bit subfolder; filenames are descriptive only because Stage 2 recognizes the format from its content. CURATOR public packages include the eight individually allowlisted CURATOR-created authority DATs: the Atari 8-bit Preservation DAT, six Atarimania GameBase DATs for Atari 2600, 5200, 7800, Jaguar, Lynx, and ST, and the Atari Lynx Dragnet supplemental DAT; all other DAT payload remains user-supplied and excluded. Sets from GameBase-derived DAT sources are placed beneath `<Output>\<Company>\<Platform>\GameBase\`. Category, revision and alternate-dump subfolders follow that bucket. Existing output is not automatically reorganized into the new bucket.
 3. Run Prepare DAR, or, after database initialization, run Stages 1, 2, and 3 separately from the full menu. Stage 2 must identify the A8P profile, and Stage 3 must publish its sets and members into Core.Database.
 4. When the export changes, replace that authority file, rebuild Stages 1-3, and retain the source date as provenance for the new authority baseline.

Use the available authoritative sources when evaluating Atari 8-bit coverage. Absence from No-Intro, TOSEC, Libretro, or MAME alone does not establish that a file is unknown, invalid, or corrupt.

Acceptance checks

- Stage 2 reports content-based A8P recognition rather than a filename-only guess.
- Stage 3 publishes stable authority-set and membership identities, including case-distinct set names and repeated same-name members.

- The committed authority revision records the export through the normal relational DAT pathway; no alternate Stage 3 route is used.

Proprietary Optical-Image Handling

CURATOR attempts supported DiscJuggler CDI, Nero NRG, and Alcohol MDF/MDS decoding during Scan Preparation. Successful decoding produces deterministic raw long-sector track members for ordinary exact-hash matching; it never establishes identity by filename or conversion alone.

The original optical-image source remains intact. Decoder output is additional matchable evidence and is authoritative only when its content hashes match the loaded DAT authority.

If an image is unsupported, malformed, incomplete, or incompatible with the bundled decoder, CURATOR logs the problem and moves on through its standard container-handling workflow. The preserved source may remain unmatched and appear in review; a decode failure is not proof of corruption.

When this changes	Required action
Platform XML/reference files	Provide or update local platform XML/reference files, then run Prepare DAR or Stages 1-3 before later-stage use.
DAT files or DAT coverage	Provide or update local DAT files, then run Prepare DAR or Stages 1-3 before curation runs.
Authority ranking or DAT profile rules	Run Stages 1-3 and validate downstream Matching/DAT Validation.
Only Source files changed	Stages 1-3 can usually be reused; start from File Discovery when the committed authority revision remains valid.
Only Destination changed	Earlier stages may be reusable; rerun Organizer only when Stage 9 evidence still matches the output plan.

8. Run Operations

A Gateway selection shares one RunID across its selected stages 1–10. A new selection receives a new identity; reused prerequisite publications retain their provenance. Collection runs require Clear or Append and recovery decisions before handoff; Prepare DAR runs only Stages 1–3 without a destination choice. Resume preserves the interrupted identity and completed output. Stage 9 freezes the plan; blocking findings prevent Stage 10 output. The active handoff uses:

-  Persistent authority database\CURATOR.db — maintained by Stages 1-3
-  Frozen Stage 9 plan — stored relationally and executed by Stage 10
-  Reference data — Organize uses the saved DAT reference library. There is no interactive DAR-file selection step.

Selected-stage runs are for targeted work. Use them only when the required persistent authority revision, prior relational stage state, source data, and workspace assumptions still match the intended operation.

Keep the normal progress display visible when monitoring a run.

9. Stage 5 and Organizer Recovery Administration

Collection recovery: choose **Clear** or **Append** before processing. **Clear** empties the selected output; **Resume** preserves completed output and continues the saved run. If required earlier results are missing, **CURATOR** stops instead of creating final output. **A completed plan cannot run again. Keep the database, saved run state and workspace together until processing finishes.**

Stage 5 Scan Preparation owns durable recovery evidence under:

Recovery authority: database\CURATOR.db, owned by Core.Database.

Active recovery: committed relational recovery or execution state, read through Core.Database.

 Completed-stage evidence relational Stage 5 state — not a recovery trigger

 **Workspace residue** workspace — not a recovery trigger

Stage 5 preserves system-looking names, frontend-style paths, hidden files and named empty archive entries for downstream content matching and membership evaluation. Names, extensions and hidden status alone do not prove that a file is dispensable. Embedded archives retain their immediate-parent lineage for recursive validation.

A raw unmatched archive container with traversed child lineage is traversal-only evidence, not a NeedsAttention artifact. Its descendants remain independently eligible for matching and review. This prevents redundant container review copies and accidental .zip.zip output names.

Prompt choice	Admin meaning
R Resume	Rehydrate committed prepared records, carry warning evidence forward, skip committed work units, retry in-flight work, and suppress the normal workspace wipe.
D Discard	Archive active recovery as discarded and start Stage 5 clean.
V View details	Show run-state ID, original RunID, checkpoint path, and FileDiscovery hash.
L Leave for later	Stop without changing active recovery state.
C Cancel	Stop without starting stage work.

Recoverable Stage 5 state permits Stages 1-3. Starting Stage 4 File Discovery prompts the operator to resume Scan Preparation and skip File Discovery, or discard recovery and start File Discovery. Stage 5 requires a recovery decision, and Stages 6-10 remain blocked until recovery is resolved. BlockedRecovery is unusable and must be archived before Stage 5 starts cleanly.

10. Maintenance Procedures

ADMIN WARNING - CLEANUP CAN REMOVE EVIDENCE

Use option H only after confirmation to clear supported history, temporary, and workspace contents while preserving required folder surfaces. The dedicated json folder and its archive/clear action are retired. Maintenance does not replace relational database maintenance.

- Do not clear _temp, workspace, stage folders or run_state while a stage is active.
- Do not treat run_state as temporary scratch.
- Do not preserve workspace as long-term evidence; preserve JSON, logs, run_state history, and final reports.
- Keep the input and workspace unchanged until Organizer finishes. Organizer uses the accepted output plan and does not recheck source contents after planning. Choose Resume to continue after the last saved output. After completion, retain the database, logs, run-state history, collection manifest, reports and final output. Stage 10 retains workspace contents until output creation and collection catalogue publication finish, then clears the contents once while preserving the workspace folder. If publication or final cleanup fails, choose Resume to finish the remaining work without creating a duplicate collection generation. Final cleanup announces the total file and folder count before deletion and completion afterward, without a cleanup progress counter. Execution.Workspace.CleanOnFinish must remain true.
- When Stage 5 recovery is active, let the recovery prompt decide whether workspace is preserved or wiped.

11. Evidence, Logs, and Validation

Evidence type	Location	Use
Relational stage state	database\CURATOR.db	Primary stage-to-stage authority, state, and summary data.
RUN.log	numbered stage folder	High-level milestones and stage timing windows.
DETAIL.log	numbered stage folder	Readable operational breadcrumbs.
DEBUG.log	numbered stage folder	Heavy diagnostics, progress rows, per-file traces, and stage telemetry.
run_state	run_state	Gateway launch/status evidence; recovery authority is stored in database\CURATOR.db.
Run Warnings	gateway summary	Stage 5 warning evidence surfaced after Stage 10.
Missing reports	Configured reporting folder (default 11_Reporting)	Company-specific report artifacts named CURATOR_<Company>_Missing_Report.txt; viewed through Reporting option R after reconciliation.
Missing report accounting	Relational execution summary; Reporting company report artifacts	Relational execution receipts carry Organizer totals and failures. Reporting-folder company reports carry full missing-item detail. Unofficial DAT-matched output counts as collected.

Gateway final timing summaries use stage RUN.log execution windows for stage rows and totals so final-screen review, missing-games pagination, and any Gateway final-summary acknowledgement time are not counted as pipeline execution time. Non-Organizer stage completion screens return directly to Gateway control without a Gateway navigation prompt.

12. Stage 10 and Output Administration

Stage 10 processes copy-only outputs first, then single-file ZIPs (including one-member sets), multi-file ZIPs, and remaining conversions. Each ZIP group shows one completed-file count bar for outputs requiring creation. ZIP output uses native 7-Zip Normal compression, level 5, with a default ceiling of 24 archive workers and one native thread per archive.

DATA-LOSS WARNING - DESTINATION POLICY

Organizer executes the current output plan. To change placement decisions, prepare a new plan through the Gateway before creating output.

- Official ordinary ROM/file outputs are individual ZIP artifacts.
- Supported disk and tape media use separate ZIPs per media image, retaining authoritative member filenames. If a required image is absent, available images go under Incomplete Sets. RAW grouping and complete multi-member cartridge handling have limited support.
- Complete Atari Jaguar CD authority sets publish as one ZIP containing every authoritative member. Post-Stage 4 planning and generation normally consume Workspace or CURATOR-owned temporary material under the frozen Stage 9 plan. Already-valid source CHDs are the deliberate exception: when a CHD satisfies the accepted authority set, Stage 10 copies it directly from the original configured Input folder instead of regenerating identical content. Stage 10 resolves the expected Input-relative path and requires it to match the recorded physical source-container path before the copy is permitted; a mismatch fails closed. The copy leaves the original CHD unchanged and reduces unnecessary CHDMan work, so the configured Input must remain available and unchanged through Stage 10. Other complete optical authority sets publish as created .chd files. A complete optical set with one safely derived structural CUE remains eligible for Official output only when every non-CUE membership is present and exact; CURATOR records the descriptor discrepancy without claiming that its bytes matched the DAT CUE. Source container type never determines the final optical format.
- Official RVZ outputs are direct .rvz files with DAT-authoritative basenames and are not zipped.
- NeedsAttention and Unofficial carry-forward outputs are evidence artifacts. Optical/container carry-forward evidence is copied directly, including CHD, RVZ, ISO, CUE, and optical BIN payloads; small non-optical loose-file evidence is ZIP-packaged.
- Append uses output discovery and retained database receipts to skip known unchanged targets without repeating content hashes or archive inspection in Stage 10. Keep the output folder unchanged during the run. Ordinary Append does not detect

same-size external content changes or changes after discovery; use the supported full physical audit after manual changes.

- Ordinary archive, copy, or conversion failures are recorded while remaining outputs continue. Resume retries unfinished work and preserves successful outputs. Catastrophic failures can stop processing; review the reported failure and recovery state.
- Execution receipts are retained relationally in Core.Database. Preserve the database and the output file 00_Manifest\CURATOR_Collection_Manifest.json.
- Collection statistics are obtained by running Reporting. Core.Database supplies the current permanent Official and Unofficial collection generation; needs_attention is a separate review queue excluded from collection enumeration, reconciliation, hashing, counting, auditing, caching, ledgers, diagnostics, viewers, and publication. After startup, Reporting announces the latest Organizer generation's new-item total and deterministic platform breakdown when additions exist; zero additions produce no new-item announcement. Reporting writes retained summaries, findings, and company reports with Platform, Expected, Retail, Unofficial, Collected, and Missing totals. Full Audit performs the slower physical SHA-256 verification, Reconcile prepares controlled recovery actions, and Detailed Export republishes retained reporting evidence.

13. Security and Data Safety

DATA-LOSS WARNING - SOURCE AND DESTINATION PROTECTION

- Back up source collections before large processing runs.
- Use copies as Source/Input, not originals.
- Use Append when existing collection contents must be protected and valid receipt evidence is available, then run Reporting reconciliation before trusting collection statistics.
- Use Clear only when the configured collection folder can be replaced; Clear is a replacement decision, not a preservation mode.
- Do not store secrets in config files, operator overrides, logs, or generated JSON.
- Do not manually edit generated JSON files.
- Keep third-party DAT/platform/tool provenance clear for traceability.

14. Troubleshooting Reference

Symptom	Likely cause	Admin action
Gateway will not start	Missing config, core module path issue, PowerShell problem, or startup notice file issue.	Confirm project structure, config, PowerShell version, and README safety notice.
Manage folders rejects paths	Path unreachable or Source/Destination/Workspace collision.	Choose reachable folders and keep the three runtime folders distinct.
Stage reports missing DAT/platform data	Authority inputs missing or misconfigured.	Restore files or update config path, then run Prepare DAR.

Symptom	Likely cause	Admin action
Recovery prompt appears	Interrupted Stage 5 recoverable state exists.	Resume, discard, view details, leave for later, or cancel based on evidence need.
Stage 4 File Discovery prompt or downstream blocked	Active Stage 5 recovery exists. Stage 4 prompts for resume/discard; Stages 6-10 require recovery resolution.	Choose Resume to continue Scan Preparation and skip File Discovery, or Discard to start File Discovery cleanly.
Archive progress remains unchanged	The active provider operation may be listing, extracting, or creating archive content.	Follow the action-specific TaskBar; if both activity and progress remain unchanged, review stage logs.
Bad archive payload warning	One archive payload failed but tooling is available.	Review ScanPreparation warnings; remaining archive work can continue.
Required tool missing	Required tool unavailable or unstartable.	Restore the configured tool before rerunning.
Physical output failure recorded	An output attempt failed. Ordinary output failures allow remaining outputs to continue; catastrophic failures can stop processing. Review the recorded failure and use Resume for unfinished work.	Review relational Organizer execution failures and the final report.
Gateway lifecycle does not resume after a child stage	Gateway/Core lifecycle state did not return after the child process exited.	Exit safely, reopen PowerShell, and review Gateway/Core logs before retrying.

Antivirus may interrupt processing

Antivirus and other security software may interrupt myOrganizer: CURATOR by temporarily blocking access to files it needs. This can cause delays or prevent an operation from completing. If this happens, review your security software's alerts and activity history. An access error alone does not confirm that antivirus software caused it.

DolphinTool warnings

If DolphinTool displays a blocking Warning dialog, read its message before dismissing it with OK. A warning alone does not establish its cause.

15. Admin Checklists

Before a full run

- Confirm PowerShell and required tools are available.
- Confirm DATs and platform files are the intended authority set.
- Confirm operator overrides and folder distinctness.
- Check for active Stage 5 recovery state.

- Choose Clear or Append in the Gateway before processing starts. Resume preserves the interrupted choice and completed output. Review the stage and Gateway completion summaries.
- Review the configured folders and recovery decision before starting the full run.

Before a selected-stage run

- Confirm required prior relational stage state exists.
- Confirm upstream relational state still matches the intended authority revision and Source files.
- Confirm Workspace assumptions for Stage 5/6 work.
- Confirm active recovery does not block the requested stage.
- Preserve retained database authority and recovery evidence required by the selected stages.

After a run

- Read the completion summary and any Run Warnings.
- Inspect stage logs in this order:

 **First** RUN.log

 **Then, as needed** DETAIL.log

 **Finally, for diagnostics** DEBUG.log

- Confirm JSON summaries and output counts.
- If Organizer ran, inspect CURATOR_<Company>_Missing_Report.txt in the configured reporting folder and any PhysicalOutputFailures.

Company Manifests, Placement Receipts, and Reporting-Owned Missing Reports

Stage 10 commits the successful Official and Unofficial generation to Core.Database and publishes 00_Manifest\CURATOR_Collection_Manifest.json from that permanent record. Organizer's receipt evidence remains the fail-closed authority for Append reuse. Company manifests may remain as administrative physical evidence, but Reporting's current collection truth comes from Core.Database and does not depend on company-presence manifests. Preserve the database, current collection manifest, and Organizer receipt evidence; deleting or altering them can break trusted continuity and require supported recovery or a deliberate rebuild.

Output Contracts

Current receipt evidence consists of the relational execution records and the exported collection manifest:

Execution receipts are retained relationally in Core.Database. Preserve the database and the output file 00_Manifest\CURATOR_Collection_Manifest.json.

i Ledger scope — The ledger may contain records for multiple companies; every record carries its Company value.

Authority is carried by the persistent relational database:

-  Persistent authority database\CURATOR.db
-  Active revision committed relational DAR authority

i Organizer authority — Organizer consumes the authority evidence embedded in the accepted Stage 9 plan. Stage 10 does not write destination-root DAR snapshots.

- CURATOR_<Company>_Missing_Report.txt lives in the configured reporting folder as pretty JSON text with a .txt extension. Reporting creates or refreshes these artifacts from the physically reconciled permanent collection generation before presenting collection coverage.
- Gateway Reporting option R requests its validated reporting and database capabilities, reads the current permanent collection generation from Core.Database, reconciles physical output and current DAR authority, and retains the run summary and findings. Summary-first company/platform coverage is displayed while item-level missing evidence remains in paired detail artifacts.

Configuration Control

- Reporting uses the configured Reporting and audit-evidence folders. Core.Database owns committed execution receipts and permanent collection state; Organizer publishes the collection manifest beneath the configured output root.
- Organizer receipts remain execution evidence. Reporting derives collection truth from the current permanent Official and Unofficial generation in Core.Database and excludes needs_attention filesystem content and NeedsAttention records from every collection boundary. Missing reports use templates\Missing_Report.template.json.
- Company authority is queried from Core.Database and is not a Reporting output artifact.

Append Behavior

- Clear replaces existing output contents for the current physical output run. Stage 10 does not write destination-root DAR snapshots or company missing reports.
- Append plans reuse from output discovery and retained database evidence. Known unchanged targets are skipped without rereading payloads in Stage 10. Different content or required member names receive separate planned destinations. Keep output unchanged during the run; same-size external changes are not detected by ordinary Append. Missing disposable needs_attention files are excluded from reuse and destination reservations and can be created again. Execution receipts retain interruption and Resume evidence; permanent collection membership is published through Core.Database.

- Always retain needs_attention as reusable source evidence, not automatic trash. Exact content can carry multiple authoritative membership edges across sets, regions, revisions, or DAT profiles. A source container may therefore produce one complete official set and a duplicate-looking review copy for another set that shares some exact members but remains incomplete. This does not indicate damage to the official artifact. Preserve the review content and include it with newly acquired material in the input source for every future run; CURATOR can then reevaluate the combined evidence and complete additional sets when their remaining members arrive. needs_attention remains separate from unofficial and outside authoritative collection truth.
- If files are manually changed outside CURATOR, use Gateway option V for a full physical SHA-256 audit. Rejected findings remain in durable audit evidence, and CURATOR does not treat manually changed files as authoritative coverage unless the operator explicitly approves reconciliation.
- Old durable review receipts are retained. Each Append consumer excludes genuinely absent needs_attention files from its working receipt map after destination-root validation. This does not delete durable receipts or weaken existing-file and permanent-collection integrity checks.

Do Not Move Generated Evidence Files

CURATOR accuracy depends on generated evidence remaining where CURATOR writes it. Do not manually move, rename, edit, or delete database authority, relational run state, placement receipts, Reporting artifacts, logs, history, or manifests during normal operation.

Protected evidence includes:

 Reporting state Reporting.json in the configured reporting folder

Execution receipts are retained relationally in Core.Database. Preserve the database and the output file 00_Manifest\CURATOR_Collection_Manifest.json.

Company missing reports: Reporting-owned files in the configured reporting folder.

 **Latest-status record** run_state\latest_status.json

 **Stage logs and history** stage History

 Persistent authority database\CURATOR.db

- Removing these files can make Append behavior, reconciliation, collection statistics, audit trails, and authority continuity inaccurate.

Global Evidence Folders

- needs_attention is a review queue outside authoritative collection truth and is excluded from Reporting collection evidence. unofficial is authoritatively matched non-retail collection output and counts toward Reporting coverage. Both remain company-routed under their global folders.

Current collection behavior

Complete Redump Sega Naomi disc sets can produce standalone CHD output using authoritative filenames. DAT-declared publication files are retained even when an image-related folder name is present. XML game identifiers remain separate from authoritative filenames.

Start collection stages through the Gateway. Clear or Append is selected before processing; Clear replaces prior output once. Resume retains the interrupted run identity and completed output without clearing again. Blocking completeness findings stop final output. Receipts are saved before generated workspace sources are removed.

Recognized catalogue categories and format folders are resolved independently. USA-inclusive sets stay at the platform root regardless of language tags; recognized alternate editions use Alternates. Original member directories remain inside ZIPs. Only a file named exactly _ with zero bytes is excluded as a placeholder.

For a bound single-member set, a missing release classification can use recognized DAT set tags such as Beta or Unlicensed. An already recognized member classification keeps precedence. Matching, payloads and set memberships are preserved.

CDI DAT matching remains unverified. No tested Dreamcast or Jaguar CD CDI has matched a loaded DAT, either as the original image or through decoded BIN tracks. CURATOR can attempt CDI decoding and preserve the original image, but successful conversion does not identify the game or confirm playback compatibility. Review unmatched originals and generated outputs before using them in your collection.

Completeness follows the selected DAT set. A DAT may describe one disk as a complete set, even when the game needs several disks. When every required member of that DAT set is found and matched, the set follows its normal destination rules.

Whole-game estimates are informational and do not determine output placement. They do not establish that a game is playable or that all editions, disks or companion files are present.

When required DAT members are missing, the available matched members remain under Incomplete Sets, grouped by their authoritative DAT set. Similar names do not combine different DAT sets. Append may add verified members to an eligible existing incomplete-set ZIP, but even a completed accumulated ZIP is not automatically moved or converted into the normal collection.

Append can add verified new members to an eligible existing ZIP in Incomplete Sets. CURATOR validates the retained archive and freezes a combined member recipe before replacing it; repeated members are skipped without rewriting the archive. Earlier contents, source history and set memberships are preserved. Completed accumulated sets remain in Incomplete Sets; automatic completion assessment and final relocation or conversion are not available.

WARNING: Preserve Incomplete Sets. Do not manually delete or move the Incomplete Sets folder or its contents. Later Append or Resume processing that needs a previously recorded incomplete-set archive will stop if that archive is missing. Recreating an empty folder is insufficient. Keep the folder and its contents in place until this limitation is addressed.

Choose Resume from the Gateway to continue the saved stage without clearing successful output. Ordinary output failures are recorded while later outputs continue; Resume retries unfinished work and preserves successful outputs. Catastrophic failures can stop processing. ZIP groups show one completed-file count bar; optical preparation uses one shared display. Collection clearing uses simple messages. Recognized cracked variants use the Cracked folder label for newly planned output; existing folders are not renamed.

Incomplete-set save details are recorded in DETAIL.log. A log file that remains unavailable can stop processing. Review the logs if cleanup reports that working files remain.

v2.5.0-beta.1 accepts existing v2.5.xx and v2.0.0-beta.6 databases at schema 26 and preserves their creation identity. This compatibility does not establish that every unfinished run can resume. Let the Gateway check recovery eligibility. Keep input and working files unchanged until processing finishes.

Stage 10 streams ZIP/Copy receipts without rechecking source content after planning or rereading fresh output. Existing Append and recovery checks remain.

Authoritative member names determine ZIP entries; configured package naming also uses authoritative set names. Naming is separate from duplicate equality. Existing CHD, CDI and RVZ formats retain their physical extensions; CHDs remain standalone and never belong inside ZIPs.

Original DAT definitions and set memberships remain preserved. Unidentified ordinary files go into folders named after their source extensions, such as A26, BIN and JPG, within needs_attention. Files without an extension remain under Unknown File. Unidentified optical discs keep their existing whole-disc review handling. Filename collisions receive numbered suffixes before the extension. Missing MD5 alone does not make a file unresolved.

Duplicate sets are compared by their complete member contents, including repeated-member counts: Size, CRC and SHA1, with MD5 compared when both declarations supply it. A missing MD5 does not split otherwise matching content and cannot bridge conflicting supplied values. Names, member order, DAT filename, format, source, platform and ranking are excluded from equality. Configured source hierarchy selects one whole-set winner; distinct sets that share only some members remain separate. Preserve every original declaration and membership. Filename comparison is case agnostic.

Unmatched review plans bind their hashes to the selected physical source, including whole CHDs and raw CDI tracks. These physical hashes are separate from normalized matching identity and from final output SHA-256 receipts.

Organizer can publish an empty permanent collection manifest. Standard Reporting and FullAudit accept empty collections and continue to report populated collections. Physical file counts and fulfilled-definition counts can differ because one physical file can retain several memberships.

RVZ creation and recovery of converted media have limited support. MAME support is very early and experimental; use clrmame to manage MAME collections. MAME output uses SPLIT sets.

Appendix A: Tool and Dependency References

ADMIN CITATION NOTE - LICENSE AND PROVENANCE RESPONSIBILITY

Citations for the current CURATOR tool and dependency set. Local configuration remains authoritative for required paths and required/optional status.

- **Authority input sources.** DAT payloads are user-supplied release-external inputs. Common upstream sources include TOSEC Downloads (<https://www.tosecdev.org/downloads>), No-Intro DAT-o-MATIC (<https://datomatic.no-intro.org/>), Redump Downloads (<https://redump.org/downloads/>), and the Libretto Database (<https://github.com/libretto/libretto-database>). Large platform XML/reference payloads should come from the administrator's own LaunchBox-style platform/library data or another explicitly reviewed local platform reference source; LaunchBox is available at <https://www.launchbox-app.com/>. Do not remove CURATOR-supplied routing platform stubs from the official release while built-in output routing rules reference them.
- PowerShell 7.6.1 minimum; PowerShell 7.6.1 or later allowed: Gateway and stage-script runtime host. Citation: Microsoft Learn: <https://learn.microsoft.com/en-us/powershell/scripting/overview>
- AlphaFS: Core.Secondary file-system services, long-path IO, UNC support, and advanced enumeration. Citation: AlphaFS GitHub: <https://github.com/alphaleonis/AlphaFS>
- HashLib4CSharp: Core.Secondary optional SHA-256 stream hashing. Citation: NuGet: <https://www.nuget.org/packages/HashLib4CSharp>
- Newtonsoft.Json: Compiled JSON serialization/deserialization path for stage artifacts. Citation: Json.NET: <https://www.newtonsoft.com/json>
- Newtonsoft.Json.Schema: JSON Schema validation dependency paired with Newtonsoft.Json. Citation: Json.NET Schema: <https://www.newtonsoft.com/jsonschema>
- Spectre.Console and Spectre.Console.Ansi: Progress Core interactive console rendering, status display, and ANSI output. Citation: Spectre.Console: <https://spectreconsole.net/> ; GitHub: <https://github.com/spectreconsole/spectre.console>
- Microsoft.Bcl.TimeProvider: Progress Core time-provider compatibility dependency. Citation: Microsoft Learn: <https://learn.microsoft.com/en-us/dotnet/api/system.timeprovider?view=net-9.0>
- 7-Zip / 7z.dll: Persistent ZIP creation, archive listing, selective mapped extraction, and Core.Archive-owned CRC32/SHA-1 streaming. Citation: 7-Zip: <https://www.7-zip.org/> ; License: <https://www.7-zip.org/license.txt>

Aaru managed libraries: Aaru.CommonTypes, Aaru.Console, Aaru.Decoders, Aaru.Filters, Aaru.Helpers, and Aaru.Images provide the in-process proprietary optical-image decoding boundary. CommonTypes is MIT licensed; the remaining bundled Aaru assemblies are LGPL-2.1-only. Citation: <https://www.nuget.org/profiles/aaru-dps> ; Project: <https://www.aaru.app/>

- Modified MAME chdman: CHD extraction, verification, and CHD artifact creation. Citation: MAME docs: <https://docs.mamedev.org/tools/chdman.html>
- DolphinTool: Required RVZ header/verify service used to read effective disc CRC32/SHA1 evidence without converting RVZ to ISO. Citation: Dolphin GitHub: <https://github.com/dolphin-emu/dolphin>

The CUE-authority resolver accepts a corrected CUE descriptor only when every extracted media member matches one unambiguous authoritative set. This check does not replace general-purpose CHD integrity verification.

- BinMerge: Optional BIN/CUE merge helper retained in the tools inventory. Citation: BinMerge GitHub: <https://github.com/putnam/binmerge> ; Releases: <https://github.com/putnam/binmerge/releases>

Appendix B: Credits, Sources, and Third-Party Attributions

CURATOR was created, architected, and is managed by Miguel A. Ortiz. Copyright © 2026 Miguel A. Ortiz. Development, coding, and documentation assistance was provided by OpenAI ChatGPT; that assistance does not transfer project ownership or third-party rights.

Bundled components remain the work of their respective authors and contributors and remain governed by their own licenses. The concise credits below supplement—not replace—the controlling upstream notices and license texts.

Component	Creator / contributor credit	License and authoritative source
Aaru managed libraries	Natalia Portillo and Aaru contributors; CURATOR modifications: Miguel A. Ortiz, with OpenAI ChatGPT assistance	MIT (CommonTypes); LGPL-2.1-only (other bundled Aaru assemblies); https://www.aaru.app/
AlphaFS	Peter Palotas, Jeffrey Jangli, Alexandr Normuradov, and contributors	MIT; https://github.com/alphaleonis/AlphaFS
HashLib4CSharp	Ugochukwu Mmaduekwe and contributors	MIT; https://www.nuget.org/packages/HashLib4CSharp
Microsoft libraries	Microsoft and .NET Foundation contributors	MIT; Microsoft.Bcl.TimeProvider and Microsoft.Data.Sqlite
Newtonsoft libraries	James Newton-King and contributors	Json.NET: MIT; Json.NET Schema: AGPL-3.0-only or commercial license; https://www.newtonsoft.com/
SQLitePCLRaw / SQLite	Eric Sink, SourceGear, LLC, Microsoft Open Technologies, Inc., and contributors; SQLite authors	SQLitePCLRaw: Apache-2.0; SQLite: public domain; https://github.com/ericsink/SQLitePCL.raw ; https://sqlite.org/
Spectre.Console	Patrik Svensson, Phil Scott, Nils Andresen, Cédric Luthi, and contributors	MIT; https://github.com/spectreconsole/spectre.console
7-Zip	Igor Pavlov; retained notices also credit Apple Inc., Facebook, Inc., Yann Collet, and Alexander Roshal (unRAR)	LGPL-2.1-or-later plus unRAR restriction and BSD components; https://www.7-zip.org/license.txt
BinMerge	Chris Putnam and contributors	GPL-2.0-or-later; https://github.com/putnam/binmerge
Modified MAME chdman	MAMEDev and contributors; CURATOR modification copyright 2026 Miguel A. Ortiz	GPL-2.0-or-later; https://docs.mamedev.org/license.html
DolphinTool	Dolphin Emulator project contributors	GPL-2.0-or-later; https://dolphin-emu.org/
Redump cue/GDI metadata	Redump project contributors	Public-domain metadata; https://redump.org/
verifydump	j68k; embedded Python, PyInstaller, OpenSSL and Microsoft contributors	Version 1.0.0; MIT plus incorporated runtime licenses; see LICENSE

Complete packaged records:  **LICENSE** and  **config\dependency-manifest.json**. The dependency manifest records the exact packaged versions, hashes, provenance, and license identifiers.

Authority and data-source credit: No-Intro / DAT-o-MATIC, Redump, TOSEC, Libretro Database, MAME, progetto-SNAPS, LaunchBox, and other operator-selected authorities remain credited to their respective projects and contributors. Their names and trademarks belong to their respective owners; mention does not imply affiliation, sponsorship, or endorsement.

Modified chdman and source availability

CURATOR bundles tools\chdman.exe as the unofficial modified MAME chdman build 0.289-CURATOR.8-poc. It retains standard chdman commands and adds logical-track hashing. Scan Preparation decodes and hashes logical tracks without writing BIN files and retains bounded header prefixes for Matching. Applicable normalization rules and pending output operations still extract tracks through the configured Core tools. Collection Assembly uses recorded relational source evidence without probing or rehashing collection media. Copying an unchanged whole CHD does not require track extraction. Acceptance is bounded to the documented collection; it does not establish universal compatibility or playback correctness. When recovery supplies an authoritative corrected CUE, Stage 9 reconciles it against the complete originating CHD membership and suppresses only obsolete recovered-CUE review records. Genuinely unresolved source evidence remains in NeedsAttention.

CURATOR checks the bundled chdman executable against its dependency manifest. Keep the supplied executable with the application; substituting another build can prevent startup or change supported disc handling.

This v2.5.0-beta.1 release provides matching application, modified chdman, modified Aaru.Images and third-party source ZIPs, a release inventory and SHA-256 sidecars. Provide all source companions beside the application download with equivalent access. Packaged Input, Output and Workspace defaults are exactly C:\; configure appropriate destinations before use. CleanOnFinish remains true and schema remains 26.

The build is licensed under GPL-2.0-or-later, credits MAMEDev and contributors, and is not produced, sponsored, approved, supported, or warranted by MAMEdev. MAME is a registered trademark of Gregory Ember.

Release dependency records identify verifydump 1.0.0, unofficial Aaru.Images 5.4.1-curator-poc1 and unofficial chdman 0.289-CURATOR.8-poc. Json.NET Schema 4.0.1 is supplied under its published AGPL alternative (publisher limit: 1,000 validations per hour). See LICENSE for exact source assets and original terms. CDI DAT matching remains unverified. This beta does not establish universal acceptance of source formats, interruption recovery or ST/GameBase inputs. Review results and preserve originals.