PRODUCT ASSURANCE · DOCUMENT 1.0

Maintenance, continuity and self-hosting

How you can inspect, run, rebuild and maintain the open-source VoLCA engine independently.

LAST REVIEWED
2026-08-22

APPLIES TO
VoLCA engine v0.10.0, pyvolca 0.10.0, reference data bundle 3

SCOPE
Open-source engine, pyvolca and reference data

CANONICAL SOURCE
https://www.volca.run/maintenance-and-continuity/

Purpose

This document answers a practical question:

> How can you continue to run, inspect, rebuild and maintain VoLCA independently?

It brings together the open-source repository, release artefacts, build automation, operating inputs and contribution paths available for the VoLCA engine.

Status key

Status	Meaning
Demonstrated	Source, automation or an artefact can be inspected and used now.
Documented	An available procedure explains how to perform the task.

Scope

Component	What it contains	Release boundary	Evidence
VoLCA engine	Haskell LCA core, HTTP API, OpenAPI description, CLI, REPL and MCP server	Signed `v*` release procedure and one engine version	Source, release process, releases
pyvolca	Python client and optional local engine launcher	Independent `pyvolca-v*` tags; source distribution and wheel published to PyPI	Python source and compatibility, pyvolca on PyPI, Python documentation
Reference data bundle	Technical reference files used for flow names, units, compartments and geography	`data/VERSION`, distributed separately with engine releases	Release workflow

The API, CLI, REPL and MCP server are interfaces of the engine binary. Their HTTP, MCP and OpenAPI descriptions derive from a shared resource registry, and release checks detect generated-description drift.

Self-hosted autonomy

Run VoLCA independently

Status: Demonstrated

You can install a released engine binary or build the engine from source. The engine runs as its own process and can expose its HTTP API, MCP endpoint and web assets. The same process is also available through its CLI or REPL. From v0.10.0 the engine answers only on the address its configuration names, `127.0.0.1` by default, so reaching it from another machine is something you ask for rather than something you inherit.

Released installers download a versioned engine and reference-data bundle, verify both against the release checksum manifest, and install them in a versioned local directory. You can pin a specific release rather than follow the latest release.

- Self-hosted quick start
- Configuration guide
- Engine releases
- Unix/macOS installer
- Windows installer

Build from source

Status: Documented

The source build uses pinned versions for GHC, MUMPS, OpenBLAS and the CI operating-system baseline. The normal verification path is:

```
git clone https://github.com/ccomb/volca.git
cd volca
./gen-version.sh
./build.sh --test --werror
```

`build.sh` also supports clean builds, coverage and static Linux builds. CI uses the same shared build matrix for pull requests and tagged releases.

Use release artefacts across platforms

Status: Demonstrated

The `v0.10.0` release, published on 22 August 2026, provides:

- Linux amd64;
- Linux arm64;
- macOS arm64;
- macOS Intel;
- Windows amd64;
- a separate reference-data bundle;
- one `SHA256SUMS` manifest covering the release artefacts.

The release workflow rebuilds and tests the tagged source before publishing those assets. Inspect the v0.10.0 release.

Preserve configuration, source data and edited state

Status: Demonstrated

For configured databases, the authoritative inputs are the configuration file, source database archives or directories, method packages and reference files. Matrix and parser caches are derived artefacts that you can rebuild from those inputs.

Uploaded databases keep their original material in their upload directory. Edits are recorded in a replayable `journal.jsonl` beside that material rather than rewriting the uploaded source. To preserve edited state, retain the complete upload directory, not only a generated cache.

A continuity set should include:

1. the exact engine release and checksum manifest;
2. `volca.toml` and every referenced configuration file;
3. source database archives or directories;
4. method packages and mapping/reference files;
5. complete uploaded-database directories, including edit journals;
6. an export of any database that should remain portable outside VoLCA.

Release and compatibility policy

Verify engine releases

Status: Demonstrated

The documented engine release process requires a release pull request, a versioned changelog, successful multi-platform CI, a deliberate signed tag and a fresh tagged build. The release workflow verifies that the tag agrees with `volca.cabal`, packages each platform and publishes checksums.

- Release process
- Engine changelog
- Release workflow

Check engine and pyvolca compatibility

Status: Demonstrated

A wire revision is the version of the JSON messages exchanged between the engine and clients over the HTTP API. It changes when the structure or meaning of a request or response changes. The engine publishes this number as `wireVersion` on its version endpoint, so a client can check that both sides understand the same API format before exchanging data.

The engine and pyvolca have independent version numbers because they are released separately. Compatibility therefore depends on the wire revisions they share, not on matching engine and pyvolca version numbers. pyvolca checks the engine's `wireVersion` on first contact, verifies that requested capabilities are supported, and identifies when the engine uses a newer format than the client understands.

The pyvolca 0.10.0 package published on PyPI supports wire revisions 2 through 11 and requires an engine at least as new as v0.9.1. These compatibility rules are included in the installed Python package. The compatibility statement shown on PyPI is generated from the same package code.

- Current pyvolca compatibility statement
- pyvolca 0.10.0 on PyPI
- Published Python documentation

Install a versioned pyvolca package

Status: Demonstrated

A `pyvolca-v*` tag starts the Python release workflow. The workflow checks that the tag matches the version in `pyproject.toml`, builds a source distribution and a wheel, validates their package metadata, and publishes those exact artefacts to PyPI. You can then install or pin the package with `pip install pyvolca==0.10.0`.

The PyPI package contains the Python client, its types and its compatibility rules. It does not bundle a VoLCA engine or an LCA database. The optional `download()` helper can fetch a compatible released engine and reference-data bundle separately when you want Python to manage a local engine.

- pyvolca release workflow
- pyvolca release history on PyPI

Follow pre-1.0 evolution

VoLCA is in the `0.y.z` series. Wire revisions, coordinated pyvolca compatibility checks, release notes and versioned artefacts make contract changes visible and let you select a known engine/client combination.

Verification and evidence

Inspect automated controls

Status: Demonstrated

The VoLCA repository includes:

- build and test jobs across five OS/architecture targets;
- warning-as-error, formatting and HLint gates;
- parser, database, matrix, inventory, impact and API tests;
- generated OpenAPI and MCP description drift checks;
- a release precheck combining mechanical gates and independent/reference-result datasets;
- release asset and checksum verification;
- separate pyvolca tests and release automation.

The latest main-branch Build, Format and HLint runs reviewed for this document completed successfully on 18 August 2026. Live status can be checked in GitHub Actions.

Review result evidence

Engineering controls show how the software is built and exercised. For methodological confidence, you can also inspect the selected source database, method, mappings, units and coverage for each calculation.

VoLCA publishes runnable examples and bounded checks separately so their workload and interpretation remain inspectable:

- Runnable examples
- Agribalyse export interoperability check
- BAFU oracle comparison
- Architecture

Maintenance continuity

VoLCA gives you several independent continuity paths:

Capability	Status	How to use it
Inspect and fork the source	**Demonstrated**	Use the open-source Apache-2.0 repository.
Install a released engine	**Demonstrated**	Select versioned artefacts and verify their checksums.
Build and test from source	**Documented and automated**	Use the shared build script and multi-platform CI configuration.
Modify the core	**Open to contributors**	Follow the architecture, tests and contribution path.
Preserve and rebuild database state	**Documented**	Retain source inputs, configuration, uploaded directories and journals.
Reproduce the release process	**Documented and automated**	Follow the documented release procedure and workflow.

We welcome you as a contributor, tester, documentation author or bug reporter. Each contribution exercises the maintenance path and strengthens the shared knowledge around VoLCA.

- Browse the source and contribution workflow
- Report a bug or propose an improvement
- Inspect current and past releases

Security and dependencies

Inspect licensing and dependencies

Status: Demonstrated

VoLCA is licensed under Apache-2.0. The repository includes `LICENSE`, `NOTICE`, full third-party license texts and a maintained inventory covering MUMPS, BLAS/LAPACK, the in-repository MUMPS bindings and direct Haskell dependencies. The inventory explains how to regenerate the direct-dependency list from a build plan.

- VoLCA licence
- Third-party inventory

These files let you review the current documented dependency and licensing perimeter directly from the repository.

Maintain this document

This page and its PDF are generated from the same Markdown source. During an engine release, the release review checks whether changes to components, platforms, versions, compatibility, installation, persistence or dependencies require an update.

The review records one of two outcomes:

- `unchanged with evidence`; or
- `update required`, followed by regeneration and verification of the page and PDF.

This keeps the version, compatibility statement, platform list and operating procedures aligned with the referenced release.

Contact and primary evidence

- VoLCA source and issues
- VoLCA releases
- Documentation
- Architecture
- Open source
- Runnable examples
- Contact VoLCA

The web page at https://www.volca.run/maintenance-and-continuity/ is the canonical current version. This PDF is a dated snapshot generated from the same Markdown source.