

Audion Hub Manager — почему устроено так

[English](#) · [О программе](#) · [Руководство](#)

Содержание

- [Решение](#)
- [Почему отдельно](#)
- [Что из этого следует](#)
- [Решение](#)
- [Единственный источник правды](#)
- [Настоящая запись требует явного намерения](#)
- [Фильтр не может молча стать полной копией](#)
- [Запись идёт по фазам, удаление — последним](#)
- [Три способа сравнения](#)
- [Расхождения — это не успех](#)
- [Мелочи, которые однажды дорого обошлись](#)
- [Проверки, которые должны оставаться зелёными](#)
- [Чего здесь нет](#)
- [Решение](#)
- [Что это значит на практике](#)
- [Исключение, которое исключением не является](#)
- [Техническое приложение: ключи профиля](#)

Три решения, определившие программу. Каждое принято не из красоты, а после случая, когда иначе было бы хуже.

1. Три слоя вместо одной большой папки

Решение

Не складывать всё в одну огромную папку документации. Проект живёт в трёх слоях, которые могут лежать под общим корнем, но остаются отдельными по смыслу.

Программа	самостоятельное портативное приложение
Зеркало	техническая отфильтрованная копия с Git
Документы	нейтральная папка заметок и описаний

Почему отдельно

Программа — инструмент. Её саму можно вести в Git и зеркалировать, как любой другой проект, но она не должна зависеть от того зеркала, которым управляет.

Зеркало техническое. Внутри — отфильтрованные копии проектов: описания, спецификации, документация, настройки, исходный код, запускающие файлы. Это слой для редактора, сравнения, коммитов и повторного использования кода.

Документы нейтральны. Слой документов не должен сканировать тысячи файлов исходников и скриптов. Там Markdown, указатели, обзоры проектов, ежедневные заметки и ссылки.

Разница в том, чем их открывают. Документы синхронизируются чем угодно — редактором, файловым менеджером, Obsidian, LogSeq, облаком или ничем. Зеркало ведётся через Git и на телефон не нужно.

Что из этого следует

Слой документов не хэшируется отдельно при сверке: он читающий, а каноническая техническая копия уже в зеркале. Сверка идёт только между исходником и зеркалом.

2. Зеркало, которое не может испортить исходник

Решение

Правила ниже перенесены из Disk Auditor — они собраны по следам настоящих поломок, и повторять их заново не нужно.

Единственный источник правды

Полный проект — единственный источник правды. Зеркало производно: оно вправе удалять и пересобирать свои файлы, но исходник не изменяет **никогда**.

Исходник при этом может быть обычным рабочим деревом Git — зеркалу нет до этого дела, оно читает только то, что разрешил профиль. Зеркало тоже может быть деревом Git, и тогда его `.git/**` защищён от сканирования, удаления и обслуживания.

Отдельно от зеркала работает редактор: он читает выбранные файлы и пишет только тогда, когда вы явно нажали сохранение. Проверочные прогоны и автоматические пробы окна исходник не трогают.

Настоящая запись требует явного намерения

Профиль может объявить предпросмотр поведением по умолчанию, и командная строка не вправе это проигнорировать:

<code>--mirror-apply ПРОЕКТ</code>	предпросмотр, если профиль так велит
<code>--mirror-apply ПРОЕКТ --apply</code>	настоящая запись
<code>--mirror-apply ПРОЕКТ --dry-run</code>	принудительный предпросмотр

Фильтр не может молча стать полной копией

Профиль обязан объявлять, что фильтры требуются и сколько масок минимум. Если масок нет — планирование падает, а не собирает полный дубль проекта.

Это верно и для будущего режима полного зеркала: если зеркалирование включено, а списки масок пусты, планирование должно упасть, пока профиль явно не разрешит обратное отдельным признаком. Ни один поставляемый профиль такого признака не ставит.

Запись идёт по фазам, удаление — последним

- 1 создать разрешённые каталоги
- 2 скопировать и обновить файлы
- 3 проставить времена изменения
- 4 если ошибок и расхождений нет — удалить лишнее в зеркале
- 5 убрать опустевшие каталоги
- 6 расставить маркеры сохраняемых пустых папок

Удаление не начинается, пока копирование не завершилось успешно. При сбое любой из ранних фаз удаление пропускается, и отчёт это прямо называет.

Настройка, прежде позволявшая удалять после ошибок, больше не действует: это ровно тот случай, когда зеркало теряет файлы, существующие только в нём.

Три способа сравнения

способ	как сравнивает	когда
быстрый	размер и время	черновая сверка
безопасный	размер и время, суммы — только при совпадшем размере и разном времени	обычная работа
строгий	суммы для всех файлов одного размера, даже при совпадшем времени	важные точки

Безопасный способ быстр, но намеренно не ловит случай «тот же размер, то же время, другое содержимое». Для контрольных точек нужен строгий.

Суммы всегда помечены алгоритмом: `b1ake3:...` или `sha256:...`. SHA-256 — запасной вариант для урезанных сред, а не равноценная замена.

Расхождения — это не успех

Любое настоящее применение с ошибками или расхождениями возвращает ненулевой код выхода. Будущая двусторонняя синхронизация обязана вести себя так же.

Мелочи, которые однажды дорого обошлись

Имена отчётов с микросекундами — иначе быстрая автоматизация перезаписывает собственные отчёты.

Область зеркалирования объявляется явно. «Отфильтрованное» означает, что цель приводится к источнику только по выбранным маскам; файлы вне области планирование не трогает.

План перепроверяется перед записью. Устаревший или отредактированный план мог удалить по пути, пересекающемуся с источником, — теперь правила и запрет пересечения проверяются заново.

Проверки, которые должны оставаться зелёными

- профиль требует масок;
- строгий режим ловит одинаковый размер и время при разном содержимом;
- безопасный режим сохраняет быстрый путь;
- удаление пропускается при сбое копирования;
- пустые служебные каталоги переживают зеркалирование.

Чего здесь нет

Двусторонней синхронизации с разбором расхождений, самоисключения слепков, политики очистки перед выпуском. Это принадлежит Disk Auditor или будущим частям; если их перенесут сюда, правила выше должны переехать вместе с ними.

3. Программа не хранит пароли

Решение

Hub Manager не становится хранилищем токенов и паролей. Программа портативна, а учётные данные Git внутри портативного проекта не живут.

Вход делегирован обычным средствам:

- ключи SSH и ssh-agent;
- диспетчер учётных данных для HTTPS;
- командные утилиты GitHub и GitLab;
- VS Code и GitKraken — для ручного входа, разбора конфликтов и наглядной работы.

Программа выполняет настоящие команды `git` и показывает их вывод целиком. Если вы уже вошли через любое из перечисленного, отправка и приём работают, а программа не знает ни одного секрета.

Что это значит на практике

Удалённый репозиторий — это всего лишь именованный адрес:

```
git remote add github git@github.com:audion/Audion_Hub.git
git remote add gitlab git@gitlab.com:audion/Audion_Hub.git
git remote add local_nas file:///Z:/git-mirrors/Audion_Hub.git
```

Проверка входа опознаёт вид адреса и подсказывает, чем именно вы будете опознаны, — но остаётся вспомогательной панелью, а не редактором учётных данных.

Исключение, которое исключением не является

Для своих серверов Forgejo и Gitea программа принимает личный токен доступа: проверяет его запросом к серверу и передаёт вашему помощнику учётных данных. В файлы проекта токен не попадает, и дальше `git push` берёт его из хранилища сам.

Токен вводится в поле и работает сразу; отдельная кнопка «запомнить» сохраняет его между запусками — и это ровно то, что обещает название. Забывание убирает токен в обоих местах: из

поля и из хранилища.

Известные адреса серверов лежат в настройках открыто — без токенов.

Техническое приложение: ключи профиля

Для тех, кто правит профили зеркалирования.

ключ	что делает
<code>dry_run_default: true</code>	предпросмотр по умолчанию; командная строка не вправе это обойти
<code>require_include_filter: true</code>	планирование падает, если масок нет
<code>min_include_globs: 1</code>	сколько масок минимум
<code>allow_unfiltered_full: true</code>	единственный способ разрешить полную копию; ни один поставляемый профиль его не ставит
<code>mirror_scope: "filtered"</code>	цель приводится к источнику только по выбранным маскам
<code>mirror_scope: "full"</code>	не применять к проектам без отдельного разбора
<code>delete_after_successful_copy: false</code>	остался в настройках, но защиту больше не отключает
<code>delete_phase_skipped: true</code>	появляется в отчёте, когда удаление пропущено из-за сбоя

Способы сравнения: `quick`, `safe`, `strict`. Значение по умолчанию на уровне кода — `metadata_then_blake3`, псевдоним строгого сравнения одинаковых по размеру файлов; прежние имена настроек сохранены. Поставляемым профилям следует держать явный `strict_blake3`, чтобы контрольные точки ловили расхождение при совпавших размере и времени.

Имена отчётов и временных файлов: `ГГГГММДД_ЧЧММСС_микросекунды`.

Любое применение с ошибками или расхождениями возвращает `exit_code: 1`.

Проверки, которые должны оставаться зелёными

```
test_profile_requires_include_masks
test_strict_mode_detects_same_size_same_mtime_different_content
test_safe_mode_keeps_disk_auditor_fast_path_same_size_same_mtime
test_delete_phase_is_skipped_when_copy_phase_fails
test_projection_mirror_preserves_empty_service_dirs
test_projection_preserves_empty_dirs_with_gitkeep
```

Маркеры пустых папок

После зеркалирования по корню проходит обслуживание: маркер ставится в каждую пустую папку зеркалируемой структуры, снимается из папок, где появились настоящие файлы или подпапки, и не ставится в скрытые технические каталоги. Маркер считается порождённым файлом зеркала, а не содержимым проекта.