

Workbench и GUI

GUI Audion Disk Tools - рабочая панель над CLI. Он создан для повторяемой файловой работы: выбрать маршрут, собрать параметры, увидеть команду, получить live log и отчёт.

Запуск

```
launcher_gui.cmd
```

Desktop wrapper выбирает свободный локальный порт, если базовый **8080** занят.

GUI с терминалом разрешён только на loopback-хостах **127.0.0.1**, **localhost** или **::1**. Non-loopback host требует явного **AUDION_ALLOW_REMOTE_GUI=1** и считается опасным remote-control режимом. Desktop wrapper не поднимает UAC сам: для осознанного запуска с правами администратора используйте **--elevate** или **AUDION_ELEVATE=1**.

Fallback без desktop shell:

```
runtime\python.exe system_core\ui_nicegui\window.py --browser
```

Рабочий маршрут

Верхняя панель содержит **Источник** и **Назначение**.

Источник используется для:

- manual sync/backup/compare;
- списка файлов;
- архивирования;
- network transfer;
- RClone upload/check.

Назначение используется для:

- manual sync/backup/compare;
- архивов;
- network transfer;
- RClone download.

Пути можно выбрать через Windows folder picker или взять из истории. Кнопка **Добавить файл...** назначает один файл текущим **Источником**, не копируя его в проект. Закреплённые записи идут сверху. При запуске GUI Workbench стартует с проектных **input/output**; выбранные внешние маршруты живут в локальной истории, а не в portable-настройках.

История путей

Файл: **config\path_history.json**.

Лимит: 100 записей.

Действия:

- Иконка pin - закрепить или открепить текущий путь.
- Open folder - открыть путь в Explorer.
- **Сбросить** - вернуть Workbench на project **input/output**, файлы не удаляются.
- Иконка delete в строке пути - удалить выбранный файл-источник либо очистить содержимое выбранной папки; внешний **Источник** требует отдельного подтверждения.
- **Удалить** - после подтверждения удалить содержимое текущих **Источника** и **Цели**, включая внешние маршруты.

Windows picker запускается в отдельном process Job Object. При закрытии приложения незакрытый picker и его PowerShell-процесс завершаются вместе с GUI; второй picker одновременно не открывается.

Дерево операций

ROOT-уровень организован по частоте:

1. **ТРАНСФЕР ДАННЫХ** — папка, шара, сервер и облако наравне; список машин, пять операций, движок под пару выбирается сам

2. **ПАПКА В ПАПКУ** — прогон самого аудитора по паре: сверка BLAKE3, карантин, отчёт о расхождениях
3. **Маски**
4. **Сохранённые профили**
5. **Архивация**
6. **Manifest / Diff**
7. **ПОДГОТОВКА**
8. **Обслуживание**

Пунктов было одиннадцать. **BACKUP-MIRROR**, **One-Way sync** и **Two-Way sync** звали один и тот же сервис через одни и те же поля и отличались единственным словом — стали одним пунктом с кнопками операций. **ОПЕРАЦИИ ПО СЕТИ** и **ОБЛАЧНЫЕ ОПЕРАЦИИ** делили одно и то же занятие по инструменту, а не по задаче — стали **ТРАНСФЕР ДАННЫХ**.

Команда выбирается слева, параметры появляются под деревом, результат виден справа.

Manifest / Diff использует текущий Workbench-маршрут **Источник -> Назначение**: можно оставить расширения пустыми для полного расчёта или собрать фильтр и применить Diff зеркалом либо односторонним прогоном.

Правый терминал

Терминал не является интерактивным shell. Это read-only live output поверхности GUI и CLI.

Поддерживается:

- UTF-8 и кириллица;
- ANSI colors через безопасный HTML-render;
- построчное обновление без полного перерисовывания;
- перенос длинных строк без горизонтальной прокрутки;
- копирование мышью;
- до 2000 строк текущей операции.

Полные логи сохраняются в **logs**. Отчёты и JSON summaries сохраняются в **report**.

Командная строка в правой панели

Файл истории: **config\terminal_commands.json**.

Особенности:

- поле команды стартует пустым;
- прошлые команды выбираются из history dropdown;
- pinned commands сохраняются отдельно;
- команды с маркерами секретов (`token`, `secret`, `password`, `Authorization`, `Bearer`, `p...`) не сохраняются и не закрепляются;
- `History` очищает незакреплённую историю;
- `Cache` полностью сбрасывает history/pinned/last command, но сохраняет shell и CWD.

Tooltip

Все `.tooltip(...)` в GUI используют единый тайминг:

- появление: `1500 ms`;
- скрывание: `100 ms`;
- transition duration: `100 ms`.

Реализация находится в `system_core\ui_nicegui\app.py`: метод NiceGUI `Element.tooltip` патчится один раз при старте приложения. Это значит, что любые новые кнопки с `.tooltip(...)` автоматически получают то же поведение.

Нативный browser `title` не используется, чтобы не появлялись два tooltip одновременно. Вместо него выставляется `aria-label`, а видимый tooltip остаётся единым Quasar/NiceGUI-слоем. Визуальный стандарт: скруглённый tooltip с фоном `RGB(23, 33, 43)`, тонкой светлой рамкой, тенью и компактным текстом. Такой стиль выбран как общий для RClone, Workbench, network-mode плиток и остальных управляющих элементов.

Сетки расширений

Большие наборы расширений в `Маски`, manual modes и сохранённых профилях строятся одинаковой карточной сеткой.

Правила ширины:

- три плитки в строке занимают одинаковые трети;
- две плитки в строке делят всю строку пополам;
- одиночная или очень длинная группа занимает всю строку;

- группы на **20+** расширений разворачиваются на всю ширину.

Внутри плитки 1-3 чекбокса остаются вертикальным списком. Начиная с 4 чекбоксов список делится на две внутренние колонки. Это удерживает вертикальные оси, убирает случайные узкие карточки и делает блоки **Видео**, **Аудио**, **Разработка**, **Картинки** визуально предсказуемыми.

Кэши и pinned UX

В GUI повторяется один паттерн:

- path history для source/target;
- terminal command history;
- mask cache;
- extension selection cache;
- RClone command cache.

Пользовательская логика везде одинаковая: закрепить, открепить, удалить, выбрать из истории, очистить поле. Это снижает когнитивный шум при длинных командах и большом числе параметров.

Трёхэтажная визуализация RClone

RClone-команды могут быть длинными, поэтому preview показывает:

1. **EXE** - найденный или ожидаемый **rclone.exe**.
2. **ROUTE** - режим и логический маршрут.
3. **SOURCE** / **TARGET** - endpoint-ы, если режим работает через универсальный маршрут.
4. **LOG** - путь log file.
5. Textarea с полной командой в Windows-quoted виде.

Команда выполняется не этой строкой, а списком аргументов subprocess. Preview нужен для визуального контроля и копирования.

У RClone есть парные toggle-панели **SOURCE ENDPOINT** и **TARGET ENDPOINT**. Их основной смысл - remote -> remote маршрут, например **drive:Project -> server:/home/user/backups**; текущий Workbench path выбирается вручную как endpoint type, если нужна локальная сторона. Отдельная toggle-панель **REMOTE** собирает авторизацию: SFTP через **user@host:port**,

OAuth/provider wizard для Yandex/Drive/OneDrive/Dropbox/Box/Mega, ключевой файл или ssh-agent.

Настройки внешнего вида

Темы лежат в `config/ui_colors.yaml`.

Выбранная тема и язык лежат в `config/gui_settings.yaml`.

Дефолтная тема: `code_dark`.

Ширина правой терминальной панели сохраняется локально в браузерном профиле GUI.