Audion Hub Manager — история

English · О программе · Руководство · Решения

Содержание

- 2.0.0
- 1.10.0
- 1.9.0
- 1.8.0
- 1.7.3
- 1.7.2
- 1.7.1
- 1.7.0
- Ранние версии

История нужна не ради счёта версий, а чтобы не чинить дважды одно и то же. Здесь записано, что менялось и почему.

2.0.0

Новое окно на Avalonia. Прежнее окно — локальный веб-сервер NiceGUI в рамке pywebview — заменено собранной программой `App\AudionHubManager.exe`. `Start.exe` запускает её напрямую, без консоли и без прав администратора. Сверху семь разделов: ИЗМЕНЕНИЯ, ИСТОРИЯ, ВЕТКИ, СИНХРОНИЗАЦИЯ, ЗЕРКАЛО, ПРОЕКТЫ, ОБСЛУЖИВАНИЕ; внизу — частые операции раздела с их полями. Четыре темы (VAULT, CODE, DARK, LIGHT), шрифт JetBrains Mono, граф коммитов, файловая панель в цветах Git. Проекты — колонкой слева: щелчок делает проект текущим, квадраты отмечают несколько, корень проектов выбирается у бейджа; в узком окне колонка раскрывается поверх страницы. Длинное имя или сообщение гаснет к концу, без многоточия. Папки проектов выбирает панель папок раздела ПРОЕКТЫ — с пометкой, чем каждая папка приходится реестру и Git, — а не системный диалог. Новый значок: оранжевая папка с веткой Git; в ABOUT — утверждённый логотип Audion и ссылки со значками:

tensionix.com, страница загрузки на audion.dev, репозиторий на GitHub, страница проекта на tensionix.github.io.

Реестр — в корне проектов, коммит — сразу нескольким. Реестр проектов — `hub_projects.json` в корне проектов, у каждого корня свой: сменил корень — колонка показывает его проекты. Реестр, который программа держала в `config\projects.json`, при первом запуске переезжает в корень. Меню корня импортирует реестр из файла и экспортирует его в файл с полными путями. КОММИТ ОТМЕЧЕННЫХ внизу колонки проводит каждый отмеченный проект через контрольную точку в его GIT-КОПИИ — зеркало, сверка BLAKE3, секреты, проверка профилем, add и commit с одной темой; проект, где шаг сказал «нет», останавливается с причиной, остальные идут дальше.

Каждая кнопка показывает свою команду. Все Git-операции окна собраны в одну таблицу — канон, 63 операции: команда, риск, что она делает и как её отменить, на двух языках. Подпись кнопки — команда так, как её набирают в терминале. Терминал показывает команду, по желанию — объяснение и способ отмены, вывод git в цвете и итог с кодом выхода. Опасная операция сначала спрашивает: показывает команду целиком, что сейчас не закоммичено и как отменить; `reset --hard` подтверждается вводом слова RESET.

Лаборатория. `hub_cli.cmd lab create` строит десять учебных репозиториев: первый коммит, индекс, ветки и слияние, конфликт, отставание от сервера, stash, теги, потерянный коммит, revert и reset, зеркало. Авторы и даты в них фиксированы, поэтому хэши одинаковы на любой машине. Каждая операция канона проверена тестом на свежей копии лаборатории; пять сетевых (ssh, gh, glab) — по форме команды.

Консольная дверь `hub_cli.cmd`. Всё, что окно делает вне Git, — команды с одним JSON на выходе: реестр проектов, зеркало, проверка путей профилем перед индексом и коммитом (файл вне профиля в коммит не попадёт), поиск секретов, хранилище, BLAKE3, Forgejo. Прежний `system_core\main.py` и четыре `launcher_*.cmd` убраны.

Свой PowerShell 7 — `App\AudionPs.exe`. Портативный PowerShell в `system_core\powershell` (четверть гигабайта) убран; сборка окружения, проверка CMD-файлов и чистка кэша работают на своём. Python распаковывается средствами .NET: в своём PowerShell нет модуля архивов.

Окружение вдвое легче. Пакеты прежнего окна (nicegui, pywebview, rich, tqdm, beautifulsoup4) ушли, `runtime\` — 60 МБ вместо 129. httpx теперь назван явно: его импортирует клиент Forgejo, а раньше он приходил только вместе с nicegui.

Чистка проекта больше не уничтожает программу. `cleanup_project.cmd` удалял все `*.exe` и `*.dll` в дереве — вместе с собранным окном и `Start.exe`. Теперь он их обходит и убирает

состояние окна и то, что оставляет его сборка (`Source\bin`, `Source\obj`).

`doctor.py` **отвечает честно.** Он проверяет пакеты движка, BLAKE3, канон, git и файлы выпуска и возвращает ошибку, если чего-то нет; раньше он всегда завершался успехом. `install\verify_portable_env.cmd` проверяет ещё дверь и собранное окно — по его собственному отчёту.

GIT-КОПИЯ для проекта любого рода. Профиль знает исходники, сборки и CI распространённых языков — Python, .NET, веб, Go, Rust, Java, Swift, Flutter, Unity, Godot и других — и не пускает зависимости, сборки, кэши, секреты, медиа, архивы, модели и базы данных; картинки и шрифты исходников проходят до мегабайта. Сотни масок сверяются за доли секунды; маркеров `.gitkeep` поставляемые профили не ставят, а то, что прячет `.gitignore` проекта, в коммит не предлагается. Пути к папкам внутри корня проектов реестр хранит относительными — корень переезжает вместе со своим реестром; новый репозиторий начинается с ветки проекта (`git init -b main`).

Зеркало не пишет сквозь ссылку. Обходы ИСХОДНИКА и GIT-КОПИИ шли по junction: junction внутри GIT-КОПИИ мог унести запись в ИСХОДНИК, а файл, который обход не смог прочитать, молча выпадал из плана — и его копия удалялась. Теперь ни один обход не идёт по символической ссылке или junction, каждая запись проверяет, куда на деле ведёт её путь, а непрочитанный файл — ошибка плана, которая останавливает удаления. Проверка секретов считает файл, который не смогла прочитать (его держит другая программа), ошибкой, а не чистым файлом; пакеты считают ошибки внутри каждого проекта; набранная команда судится по доводам, `push -f` или `clean -fdx` и внутри строки PS.

Документы — о программе как она есть, на двух языках. Git-вики собирает все команды по разделам окна: что делает каждая и как её отменить, четыре места Git, слова Git простыми словами, частые положения «что делать», учебная лаборатория. Новый справочник консольной команды (`CLI_RU.md`). Таблицы команд пишет из канона `hub_cli.cmd canon --docs`, и тест следит, чтобы они не отстали от кнопок.

Убраны шаги сборщика 04 (портативный PowerShell), 74–78 (лицензии и архив — их делают Audion Build Licenses и Audion Build Publisher), 90, 91 и 99; `system_core\license`, очистка шаблона, пустые `data`, `backup`, `release`; код движка, который звало только прежнее окно: HTML-терминал, Git-действия его кнопок, открытие VS Code и браузера, разбор команд прежней строки.

1.10.0

Испорченный реестр больше не мешает запуску. Одна лишняя запятая в `config/projects.json` роняла программу с трассировкой ещё до появления окна: реестр читался строгим разбором на этапе загрузки. Теперь чтение настроек при запуске откатывается к умолчаниям, сохраняет уцелевшие записи и говорит о проблеме в окне вывода. Пустой или нечитаемый реестр даёт временный проект, указывающий на саму программу, — окно открывается, и настройки правятся изнутри.

План зеркалирования перепроверяется перед записью. Применение доверяло корням, записанным в файле плана, — устаревший или отредактированный план мог удалить по пути, пересекающемуся с исходником. Профиль и запрет пересечения проверяются заново, прежде чем что-либо будет записано или удалено.

Изменение поведения: ошибка копирования безусловно запрещает удаление. Прежде настройка `delete_after_successful_copy: false` позволяла удалять после сбоев — ровно в том случае, когда зеркало способно потерять файлы, существующие только в нём. Настройка осталась в конфигурации, но отключить эту защиту больше не может.

Команды через движок, а не через строку оболочки. Создание пакета и приём перемоткой собирали командную строку в окне, подставляя в неё путь файловой системы: имя папки с `&` или кавычкой ломало команду. Теперь обе идут списком доводов. Нужные функции движка существовали всё это время и просто не вызывались.

Опознание в панели задач. Имя, уже начинающееся с марки, больше не получает второй приставки: программа зовётся `Audion.Hub.Manager`, а не `Audion.Tools.Audion.Hub.Manager`. Windows группирует окна именно по этому имени.

Новый значок — белый трёхлучевой знак на почти чёрной плитке со скруглением, в размерах от 16 до 256 точек; ниже 32 точек упрощённый вариант, где три отдельных узла сливаются. Прежний значок сохранён.

Убраны два помощника обращения к серверу, которыми ничто не пользовалось.

1.9.0

Токен заработал сразу. Прежде каждая кнопка читала токен обратно из хранилища — и введённый в поле не делал ничего, пока не был сохранён. Получалось, что сохранение было условием работы, а не удобством. Теперь введённый токен используется немедленно, а

«запомнить токен» делает ровно то, что обещает названием: помнит между запусками. Строка состояния называет, каким именно токеном воспользуются кнопки.

Забывание чистит оба места сразу — и поле, и запомненную копию в хранилище учётных данных.

1.8.0

Клонирование появилось в панели удалённых, где не было вовсе. Берёт адрес из поля, клонирует рядом с папкой активного проекта, отказывается клонировать поверх существующей папки и запускает `git` списком доводов, а не через оболочку.

Клонирование не виснет. Оно идёт с запретом на запрос пароля: запрос из окна всё равно некому увидеть, и вместо ожидания в пустоту программа сразу предупреждает, что Git читает хранилище, а не форму.

Панель удалённых помнит выбор — платформу и вид адреса между запусками, и открывается там, где её оставили, а не сбрасывается на GitHub.

1.7.3

Области действия токена следуют маршруту, а не объекту. Проверка на живом сервере Forgejo 15 показала: создание репозитория лежит под `/api/v1/user/*` и потому требует права на пользователя, а не на репозиторий. Git по HTTPS оценивается отдельно и требует права на репозиторий в любом случае — одним набором закрыть и обращения к серверу, и Git нельзя.

Записано, чем плох адрес с учётными данными: Git пишет такой токен открытым текстом в `.git/config` рабочей копии, где он переживает каждый резерв и каждое зеркало этой папки. Та же семья ошибок, что секрет в параметрах адреса, и полностью обходится через помощника учётных данных.

1.7.2

Закрыт путь внедрения команды. Разбор адреса сохраняет `;`, `&&` и `$(…)` внутри имени хоста, а это имя подставлялось в пробу `ssh -T`, которую окно запускает через оболочку. Файл `forgejo_hosts.json`, приехавший с клонированным проектом, мог таким образом выполнить команду на первой же пробе. Имя хоста, пользователь и порт проверяются там, где сходятся все пути; враждебные записи отбрасываются вместо загрузки; адреса со встроенными учётными данными отвергаются сразу.

Отвергаются адреса, заставляющие Git выполнить команду (`ext::`, `fd::`), адреса, начинающиеся с дефиса, — Git прочтёт их как ключ, — и адреса с управляющими символами. Проверка идёт и при сохранении из панели, и при применении файла настроек, чтобы конфигурация из чужого проекта не протащила такой адрес.

Предупреждение о простом HTTP перед отправкой токена: для сервера в домашней сети это законно, но токен пойдёт по проводу открытым.

1.7.1

Сервер, введённый в форму, больше не теряется. Запись настроек шла через функцию, которая выходила раньше времени, если учётная запись уже совпадала, — и хост не попадал в файл, так что при следующем запуске его приходилось вводить заново.

Разбор настроек стал устойчивым к ошибкам. Файл, поправленный руками с опечаткой, прежде ронял обработчик события: панель читает его при каждом переключении платформы и каждом нажатии клавиши в поле адреса.

Список репозиториев больше не обрезается молча. Перебор страниц останавливается на тысяче записей, и достижение потолка теперь называется вслух, а не выдаётся за полный список.

Негодный порт отвергается, а не подменяется тихо на двадцать второй — это строило правдоподобный адрес, который отказывал потом, при отправке.

Адрес со встроенным логином или токеном не сохраняется в `config/remotes.json` — это файл, попадающий в коммит, и секрет был бы опубликован следующей записью.

Удаление токена требует названной учётной записи: хранилища учётных данных различают записи по хосту *и* имени, поэтому пустое имя не совпадало ни с чем, сообщая при этом об успехе.

1.7.0

Поддержка своих серверов Forgejo и Gitea. Файл `config/forgejo_hosts.json` описывает сервера — адрес, пользователя и порт SSH, предпочтительный вид адреса, запомненную учётную запись — и не содержит секретов.

Принят обычный договор учётной записи: личный токен доступа, проверяемый запросом к серверу и передаваемый помощнику учётных данных. В настройки проекта не пишется ничего, а обычная отправка по HTTPS подхватывает тот же токен.

Вход через OAuth2 рассмотрен и отклонён: в Forgejo для него не реализованы области действия, такие токены несут административные права учётной записи, быстро истекают и сами по себе отправку не обслуживают.

403 отделено от 401 в каждом обращении. Недостаточное право теперь сообщает: «токен верный, но создан без права, которое нужно этому вызову», — называет требуемое право и предлагает рабочий набор, вместо голого кода состояния рядом с токеном, который на самом деле в порядке.

Текст ошибки сервера затирается перед показом. Forgejo цитирует неизвестный токен обратно в сообщении — и отозванный или истёкший токен появился бы в окне вывода дословно, а оттуда в снимках экрана и скопированном тексте.

Записан случай 413 для серверов за прокси или туннелем с ограничением на размер запроса: упирается в него именно первая отправка истории, а обход — отправка по SSH, в том числе через проброс порта.

Ранние версии

`0.2.x` — планирование: договоры безопасности зеркала, разделение слоёв, профили фильтрации. `v0.1.0` — скелет приложения.

Из этого периода в программу перешли правила, изложенные в Решениях: они собраны по следам настоящих поломок Disk Auditor, и повторять их заново не нужно.