

Установка и запуск

[English](#) · [Руководство](#) · [Справочник](#)

Содержание

- [Требования](#)
- [Запуск](#)
- [Сборка окружения](#)
- [Сборка окна](#)
- [Зависимости](#)
- [Папки проекта](#)
- [Обслуживание](#)
- [Перед большим прогоном](#)

Проект переносимый: он не устанавливается в систему, не пишет в реестр и целиком живёт в своей папке. Скопировать папку на другую машину — достаточная «установка».

Требования

Windows. Программа рассчитана на Windows: перенос колонок по адресу и конвертация старых `.doc` / `.xls` работают через Word и Excel.

Excel. Нужен команде «Перенести колонки по адресу»: именно через него сохраняются формулы, оформление и порядок строк основной книги. Word — только конвертации `.doc`. Остальные команды без них работают.

Интернет. Нужен один раз для сборки окружения и потом изредка — для скачивания реестра ОКТМО.

Собранный portable-проект ничего больше не требует: окно лежит в `App\` вместе со своей средой .NET, Python — в папке `runtime`. Ставить .NET или PowerShell на машину не нужно.

Запуск

Способ	Что делает
<code>Start.exe</code>	Открыть окно программы
<code>App\AudionAddressProcessor.exe</code>	То же самое напрямую
<code>address_cli.cmd</code>	Консольная команда: те же операции без окна (см. Справочник)
<code>builder_main.cmd</code>	Сборка portable-окружения

`Start.exe` запускает `App\AudionAddressProcessor.exe` и отходит в сторону. Второй запуск из той же папки не открывает второе окно, а выводит первое вперёд.

Окно находит папку программы по своему расположению (`App\` лежит в корне) и вызывает движок Python из `runtime\`: `runtime\python.exe`, затем `runtime\python\python.exe`. Все операции идут отдельным процессом, поэтому СТОП и закрытие окна не оставляют за собой ни Python, ни Excel, ни Word.

Настройки окна лежат в `config\window_settings.json` (размер, язык, тема, размер шрифта), история путей — в `config\path_history.json`. Окно при первом запуске открывается на русском; английский включается кнопкой EN и запоминается.

Если окно не открылось

Частые причины:

- **Нет `App\`.** `Start.exe` говорит, чем её собрать: `install\Build-App.cmd`.
- **Окружение не собрано.** Окно открывается, но в журнале написано, что нет Python проекта, и операции не запускаются — соберите окружение (`builder_main.cmd`, шаг `[01]` или `[02]`).
- **Окно упало.** Подробности записаны в `logs\gui-crash-ВРЕМЯ.log`, ход дня — в `logs\gui-ДАТА.log`.

Проверить связку «окно — команда — манифест» без показа окна:

```
App\AudionAddressProcessor.exe --self-check=check.json
```

Окно строит все разделы, сверяет их с манифестом и пишет в файл, сколько операций и полей нарисовано; код выхода 0 — всё на месте.

Сборка окружения

Нужна только если вы получили проект без папки `runtime` или пересобираете его. Сборка обновляет компоненты — Python и пакеты; окно (`App\`) она не трогает.

`builder_main.cmd` — меню сборки:

Пункт	Что делает
[01] PYTHON ENV CMD	Собрать portable-окружение (основной путь)
[02] PYTHON ENV PS	То же через PowerShell программы (<code>App\AudionPs.exe</code>)
[05] WINDOW	Пересобрать <code>App\</code> из <code>Source\</code> — только после правки исходников окна
[09] PORTABLE OFFLINE	Установка из локального <code>wheelhouse</code> без сети
[70] CLEAN INSTALL CACHE	Очистить кэш сборки и остатки сборки окна; собранные <code>runtime</code> и <code>App\</code> остаются
[71] VERIFY / DOCTOR	Проверить окружение, консольную команду и собранное окно

Установка без интернета

Если в проекте есть папка `wheelhouse` с колёсами зависимостей, окружение собирается из неё: `install\install_portable_offline.cmd` или пункт `[09] PORTABLE OFFLINE`.

Проверка

`[71] VERIFY / DOCTOR` — или `install\verify_portable_env.cmd` — проверяет интерпретатор и импорт всех зависимостей, что консольная команда читает манифест, что `Start.exe` и `App\` на месте и что окно рисует каждую операцию и каждое поле манифеста. Это первое, что стоит запустить, если что-то не работает.

Сборка окна

Окно — программа на Avalonia 12 и .NET 10, исходники в `Source\`. Собранное окно лежит в `App\` и едет в выпуск готовым, поэтому обычная сборка окружения его не пересобирает.

Пересобирать нужно после правки `Source\` или когда манифест получил что-то, чего окно ещё не умеет рисовать:

```
install\Build-App.cmd
```

Нужен .NET 10 SDK. Скрипт прогоняет проверки окна (`tests\test_window_avalonia.py`), собирает окно и свой PowerShell `AudionPs.exe`, проверяет собранное (`--self-check`) и только тогда заменяет `App\`: неудачная сборка оставляет прежнюю работающую копию. Номер берётся из `config\version.json`.

`Start.exe` собирается отдельно: `install\Build-StartLauncher.cmd`.

Зависимости

Пакет	Зачем
<code>pandas</code>	Табличные операции
<code>openpyxl</code>	Чтение и запись XLSX
<code>rapidfuzz</code>	Приблизительное сравнение названий улиц
<code>py morphology3</code> , <code>py morphology3-dicts-ru</code>	Падежные формы названий населённых пунктов
<code>PyMuPDF</code>	Текстовый слой PDF
<code>pywin32</code>	Работа с Excel и Word
<code>pyyaml</code>	Манифест и файлы настроек
<code>tqdm</code>	Ход длинных стадий
<code>setuptools</code> , <code>wheel</code> , <code>packaging</code>	Сборка окружения
<code>pytest</code>	Тесты

Окно от пакетов Python не зависит. Тексты лицензий всех сторонних компонентов лежат в `licenses/`.

Папки проекта

Папка	Что в ней	Можно чистить
App	Собранное окно и свой PowerShell	Нет, пересобирается из Source
Source	Исходники окна	Нет
system_core	Исходный код движка и консольной команды	Нет
config	Настройки, манифест, раскладка окна, ключи и пины ОКТМО	Нет
data	Реестр ОКТМО и справочник регионов	Реестр скачивается заново
runtime	Portable Python	Пересобирается
wheelhouse	Колёса для офлайн-установки	Да
input	Исходные файлы	Да
output	Результаты	Да
report	Отчёты прогонов	Да
logs	Журналы операций и окна	Да
workspace	Промежуточные файлы	Да
release	Архивы результатов	Да
docs	Документация	Нет
licenses	Лицензии сторонних компонентов	Нет
tests	Тесты	Нет

Первичное создание рабочих папок — `install\init_folders.cmd`.

Обслуживание

Очистка проекта. `cleanup_project.cmd` удаляет наработанные артефакты: кэши, временные конвертации, журналы, отчёты, снимок Росстата `data\rosstat` (он скачивается заново) и пересобираемые `runtime` и `wheelhouse`. Код, настройки, документация, исходники, собранное окно `App\` и утверждённые эталоны сохраняются. Внимание: очистка опустошает и `input`, и `output` — сначала заберите оттуда нужное.

Кодировка CMD-файлов. После правки любого `.cmd` запускайте `install\Check-CmdEncoding.cmd`: файлы должны остаться UTF-8 без BOM с переносами CRLF. Иначе кириллица в меню превратится в мусор.

Обновление зависимостей. `install\Update-Requirements-Lock.cmd` регенерирует lock-файл.

Реестр ОКТМО обновляется не отсюда, а кнопкой в окне — см. [ОКТМО](#).

Перед большим прогоном

Проверьте окружение (`VERIFY / DOCTOR`), работайте с копиями важных таблиц и держите источники, эталон и результаты в разных папках.

Первый прогон на новом наборе данных делайте на десятке строк, ответ для которых известен заранее.