

Smoke-Test Checklist

Содержание

- [Runtime Imports](#)
- [Syntax](#)
- [CMD Encoding](#)
- [Cleanup / Init Check](#)
- [Portable Tool Installer Check](#)
- [NiceGUI Smoke](#)
- [Server Check](#)
- [Window Check](#)
- [Picker Check](#)
- [Layout Check](#)
- [Visual Smoke Screenshots](#)
- [Nested Menu And Fields Check](#)
- [PowerShell / CLI Window Check](#)
- [NiceGUI ProcessPool Fallback](#)

Runtime Imports

```
runtime\python.exe -c "import nicegui, webview, yaml, rich; print('OK GUI imports')"
```

Если проекта ещё нет в portable runtime, используйте системный Python 3.12.

Syntax

```
runtime\python.exe -m py_compile system_core\ui_nicegui\app.py  
system_core\ui_nicegui>window.py
```

CMD Encoding

Все `.cmd` должны быть:

```
UTF-8 without BOM + CRLF
```

ВАЖНО: ВСЕ `.CMD` ФАЙЛЫ ОБЯЗАТЕЛЬНО UTF-8 БЕЗ BOM И СТРОГО CRLF.

LF-only `.cmd` перед релизом исправить. После любого patch/edit проверять `BOM=False` и `LoneLF=0`.

Штатная проверка/ремонт для проекта:

```
install\Check-CmdEncoding.cmd -Fix
```

Эта же проверка встроена в portable build, offline install, verify и release archive gate.

Cleanup / Init Check

- `install\init_folders.cmd` создаёт все управляемые рабочие папки проекта.
- `install\init_folders.cmd` создаёт `.gitkeep` в пустых воспроизводимых папках: `runtime`, `wheelhouse`, `install\download`, `Tools\*\bin`, `logs`, `report`, `workspace`, `data`, `release`.
- `install\init_folders.cmd` не создаёт пустые файлы ключей, токенов или секретов.
- `cleanup_project.cmd` повторяет рабочие зоны проекта и чистит всё воспроизводимое или сгенерированное: `runtime`, `wheelhouse`, `install download/cache/release-артефакты`, `Tools\ffmpeg`, `Tools\yt-dlp`, `Tools\deno`, `Tools\7zip`, `system_core\7zip`, `portable PowerShell/FZF`, `logs/report/workspace/data/release`, `Transcoded` и временные файлы.
- `cleanup_project.cmd` после чистки вызывает `install\init_folders.cmd`.
- `cleanup_project.cmd` не удаляет пользовательские `Source`, `Download`, `LUTs`, а также `config`, `docs`, `GitHub`, `licenses`, `launchers`, `install-скрипты` и исходный код вне явных `cleanup-targets`.

Portable Tool Installer Check

- `install\Install-Portable-FFmpeg-BtbN.cmd` ставит BtbN win64 `gpl` через `install\Ensure-7zip.ps1`, после успешной загрузки/распаковки очищает весь

`Tools\ffmpeg`, затем создаёт `Tools\ffmpeg\bin` и копирует свежий payload.

- `install\Install-Portable-FFmpeg-Gyan.cmd` требует 7-Zip, ставит Gyan Stable FULL-сборку через direct/GitHub failover, после успешной загрузки/проверки очищает весь `Tools\ffmpeg`, затем создаёт `Tools\ffmpeg\bin` и копирует свежий `bin` payload.
- `install\Install-Portable-yt-dlp.cmd` очищает весь `Tools\yt-dlp`, затем создаёт `Tools\yt-dlp\bin` и копирует свежий `yt-dlp.exe`; затем ставит Deno в `Tools\deno`.
- `install\Install-Portable-7Zip.cmd` очищает весь `Tools\7zip`, затем создаёт `Tools\7zip\bin` и копирует свежий payload.
- `install\Install-Portable-PowerShell.cmd` очищает `system_core\powershell` перед копированием свежего portable PowerShell.
- `install\launcher-tools-update_fzf.cmd` удаляет старый `system_core\fzf.exe` перед копированием нового.
- Не допускается обновление media-tools простым `copy /y` поверх старой tool-папки: старые EXE/DLL/docs и legacy-файлы должны исчезать при обновлении.

NiceGUI Smoke

```
runtime\python.exe system_core\ui_nicegui\app.py --smoke
```

Ожидается строка `OK nicegui shell`.

Эта проверка также вызывается из:

```
install\verify_portable_env.cmd
```

если `system_core\ui_nicegui\app.py` существует.

Server Check

Запустите на свободном тестовом порту:

```
runtime\python.exe system_core\ui_nicegui\app.py --host 127.0.0.1 --port 8099 --no-browser
```

Проверьте `http://127.0.0.1:8099/`.

Window Check

launcher_gui.cmd

Ожидается отдельное desktop-окно ruwebview. Браузер не должен открываться сам.

Ожидаемый стартовый размер окна: 1600x900. Минимальный размер около 1180x720.

Picker Check

Проверьте:

- Add files... открывает Windows file picker;
- можно выбрать несколько файлов;
- файлы копируются в input;
- Add folder... копирует папку в input;
- повторяющиеся имена получают уникальные suffix;
- попытка добавить сам input не проходит.

Layout Check

На WUXGA 1920x1200 с Windows scale 150%:

- окно можно сжать до ноутбучного логического профиля без развала двух колонок;
- команды остаются слева, статус и терминал справа;
- терминал не уезжает под список команд;
- нет раннего перехода в вертикальную ленту.

На FullHD/4K:

- кнопки в одну строку;
- терминал справа занимает большую часть высоты;
- status/progress не раздувают layout;
- Cancel виден только во время операции;
- Logs рядом с терминалом.

- **Command** в шапке терминала виден и не конкурирует с главным запуском операции.
- под терминалом есть многострочная command line для админских утилит: Shell, история/пины, команда, CWD, File/Folder picker.
- под терминалом есть постоянный итоговый индикатор: серый в ожидании, синий во время выполнения, зеленый после успешного завершения, красный после ошибки.
- левая колонка не растягивается бессмысленно;
- терминал остается читаемым;
- нет пустых карточек ради декора.

Visual Smoke Screenshots

После заметных GUI/layout-правок сохраняйте smoke-скриншоты в проектной зоне отчётов, например:

```
report\gui_smoke_screenshots\
```

Минимальный набор:

- корневое меню с рабочими папками и терминалом;
- один главный экран команды с основными полями;
- тот же или похожий экран с раскрытым **Дополнительно**, если менялись редкие поля;
- экран TASK/длинной формы, если проект использует nested **operation_groups**;
- терминал и нижний статус после короткого успешного запуска.
- терминал с dry-run предпросмотром длинной команды, чтобы проверить перенос и monospace-читабельность.

Цель не в красивом альбоме, а в том, чтобы портирование ловило реальные UI-регрессии: не помещающиеся dropdown, слишком яркие рамки, дублирующие controls, ранний перенос в одну колонку, лишний scroll и невидимый финальный статус.

Nested Menu And Fields Check

Если проект использует **operation_groups** и **fields**:

- переходы по вложенному меню меняют только левую область команд;

- правый терминал, статус, прогресс и кнопки папок остаются на месте;
- leaf-команда сначала показывает финальный экран `Запустить` / `Назад`;
- `text`, `number`, `select`, `checkbox` и `checkboxes` отображаются корректно;
- `radio`, `profile_buttons` / `preset_buttons` отображаются корректно, если используются;
- `options_source` загружает динамические варианты, кэшируется и обновляется кнопкой `Обновить список`;
- `checkboxes` переносится на несколько строк и не ломает ширину окна;
- `min_selected: 1` блокирует запуск, если пользователь снял все флажки;
- выбранные значения видны в `context.operation.parameters` или в итоговой CLI-команде;
- кнопка `Command` строит dry-run через тот же сервис, который запускает реальную операцию;
- пустые строки сохраняются, если они означают auto/default.
- схожие выборы на форме стоят рядом и читаются как один блок;
- маленький фиксированный single-choice не спрятан в dropdown без причины;
- нет двух почти одинаковых кнопок запуска для одного пользовательского результата;
- нет дублирующих controls избранного для одного и того же списка;
- нет оторванных команд без объекта: `В избранное`, `Сохранить`, `Проверить`, `Удалить` должны быть переименованы или визуально привязаны к единственному полю;
- второстепенные параметры не отодвигают основной запуск ниже видимой области.
- редкие параметры можно свернуть в `Дополнительно`, а состояние блока запоминается, если проект это поддерживает.

PowerShell / CLI Window Check

Если проект вызывает `pwsh.exe`, `powershell.exe`, Office COM helpers, ffmpeg или другой дочерний CLI:

- запуск через GUI не должен создавать всплывающие консольные окна на каждый файл;
- `-WindowStyle Hidden` не считается достаточной защитой;
- проверьте, что subprocess использует `run_process()` или `STARTUPINFO/SW_HIDE` и `CREATE_NO_WINDOW`;
- вывод дочерней команды должен идти в правый GUI-терминал или лог, а не в отдельное пользовательское CLI-окно;
- ffmpeg-команды с известной длительностью должны выводить `[PROGRESS]` строки текущего файла в терминал;

- после завершения операции зеленый индикатор под терминалом остается видимым, даже если окно было неактивно.

NiceGUI ProcessPool Fallback

В закрытых portable/sandbox окружениях NiceGUI может не создать multiprocessing process pool. GUI должен стартовать всё равно, если проект использует только обычные GUI-задачи и `run.io_bound`.

Проверка считается успешной, если `runtime\python.exe system_core\ui_nicegui\app.py --smoke` проходит, а серверный запуск не падает на `PermissionError` / `WinError 5`.

Для native picker dialogs PowerShell ищется в таком порядке:

1. `system_core\powershell\pwsh.exe`
2. `pwsh.exe` из `PATH`
3. встроенный `powershell.exe`

Наличие хотя бы одного варианта видно в секции `[GUI portability]` команды:

```
runtime\python.exe system_core\doctor.py
```