

Hardware И Диагностика

Зачем Нужна Диагностика

Media pipeline зависит не только от Python-кода, но и от:

- FFmpeg build;
- ffprobe;
- yt-dlp;
- Deno;
- драйверов GPU;
- доступности CUDA/NVENC, QuickSync, AMF/D3D11VA;
- наличия SVT-AV1 и dav1d;
- валидности manifest/profile map.

Диагностика отделяет “профиль не поддерживается на этой машине” от “проект сломан”.

Inventory

Считает рабочие папки:

- Source;
- Transcoded;
- Download;
- LUTs.

Это быстрый sanity check: видит ли проект ожидаемые папки и сколько там файлов.

Selftest

Проверяет:

- FFmpeg;
- ffprobe;
- yt-dlp;
- Deno.

Если selftest падает, сначала ремонтируй portable tools через builder/installers.

Profile Doctor

Сверяет:

- GUI manifest;
- старые `Scripts\*.cmd`;
- `config\script_profiles.yaml`;
- service bindings.

Нужен после добавления или переименования пресетов. Если GUI показывает профиль, но wrapper/runner о нём не знает, это должен поймать Profile Doctor.

Hardware Capabilities

Проверяет доступность backend-ов:

- CPU H.264/HEVC;
- CUDA/NVENC - H.264, HEVC и AV1;
- Intel QuickSync/QSV - H.264, HEVC и AV1;
- AMD AMF/D3D11VA - H.264, HEVC и AV1;
- SVT-AV1;
- dav1d AV1 decode.

Если железа или драйвера нет, результат должен быть `MISS`, а не project failure. Сюда же относится случай, когда GPU есть, но конкретный кодек ему недоступен: старая Intel-графика отвечает на `av1_qsv` строкой `Current codec type is unsupported`, и это `MISS`.

Декодирование проверяется отдельными smoke-тестами: **CUDA decode**, **QuickSync decode**, **AMD/D3D11VA decode** и **dav1d AV1 decode**. Это принципиально: декод и кодирование - разные стеки, и на машине без NVENC или AMF аппаратный декод может быть полностью работоспособен. Guard для **Декодировать через** смотрит только на соответствующий decode-тест.

Если cache создан старой версией и не содержит decode-строк, preflight выдаёт **WARN** с предложением перезапустить **Hardware Capabilities**, а не блокирует работу.

Hardware cache используется GUI для блокировки недоступных hardware backend до запуска длинного FFmpeg encode.

Preset Test Matrix

Запускает набор пресетов на коротком fixture-файле. Смысл:

- проверить сборку команд;
- проверить доступность codecs;
- отличить **MISS** от **FAIL**;
- не тратить время на длинные реальные source-файлы.

Категории обычно покрывают:

- CPU encode;
- hardware encode;
- hardware backend;
- audio;
- remux;
- FPS;
- LUT;
- storage;
- delivery.

Категория **hardware backend** существует отдельно потому, что все пресеты манифеста собраны под CPU: без неё ветки NVENC/QSV/AMF в сборщиках команд не выполнялись бы ни разу. Она прогоняет матрицу **backend × target** (cuda/qsv/amd × H.264/HEVC/AV1) прямо через рабочий сборщик команд. На машине без соответствующего железа это **MISS**.

Tool Installation

GUI содержит явные операции:

- установить 7-Zip;
- установить FFmpeg;
- установить yt-dlp.

Для полной первичной сборки всё равно удобен:

```
builder_main.cmd
```

Но GUI install page удобен для точечного ремонта повреждённого tool payload.

Как Диагностировать Проблему

FFmpeg Не Найден

1. Selftest.
2. Проверить `Tools\ffmpeg\bin\ffmpeg.exe`.
3. Переустановить FFmpeg через builder/install.
4. Повторить Selftest.

Hardware Профиль Заблокирован

1. Запустить Hardware Capabilities.
2. Проверить driver/GPU.
3. Если backend `MISS`, выбрать CPU или другой hardware backend.
4. Не считать `MISS` ошибкой проекта.

LUT Профиль Падает

1. Проверить `LUTs\`.
2. Выбрать LUT явно.
3. Нажать `Команда`.
4. Проверить escaping пути в dry-run.
5. Запустить `Тест: первый файл`.

YouTube Сломался

1. Selftest yt-dlp.
2. Обновить yt-dlp.
3. Попробовать Safe Safari.
4. Включить IPv4/retries.
5. Проверить cookies, если требуется auth.

Минимальный Health Check

После ремонта окружения:

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

В GUI:

1. Selftest.
2. Hardware Capabilities.
3. Profile Doctor.
4. Preset Test Matrix.

Если эти пункты зелёные или дают ожидаемые **MISS**, можно переходить к реальным Source.