

Media Smoke Test Checklist

Базовая Проверка Python

```
runtime\python.exe -m py_compile system_core\ui_nicegui\app.py  
system_core\services\media_service.py Scripts\Common\script_runner.py
```

NiceGUI Smoke

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

Ожидаемо:

```
OK nicegui shell: <project path>
```

Запуск Тестового GUI

Если основной порт занят, используй отдельный:

```
runtime\python.exe system_core\ui_nicegui\app.py --host 127.0.0.1 --port 8081 --no-  
browser
```

Проверить порт:

```
Get-NetTCPConnection -LocalPort 8081 -State Listen
```

UI Smoke

В браузере:

1. открыть GUI;
2. проверить Source/OUT;
3. открыть **Аудиопотоки** ;
4. увидеть segmented-кнопки:
 - **ИЗВЛЕЧЬ АУДИО** ;
 - **ЗВУК В ВИДЕО** ;
 - **КОНВЕРТИРОВАТЬ АУДИО** ;
5. задержать курсор на кнопке примерно на секунду;
6. проверить, что tooltip не обрезается на первой и последней кнопке;
7. нажать **Команда** для dry-run.

Tooltip Проверка

DOM-level признаки:

- элементы имеют **data-audion-tooltip** ;
- CSS **::after** содержит текст подсказки;
- первая segmented-кнопка имеет left alignment;
- последняя segmented-кнопка имеет right alignment.

Для первой кнопки computed style должен показывать:

```
left: 0px  
transform: matrix(... 0, 2)
```

Для последней:

```
right: 0px  
transform: matrix(... 0, 2)
```

Проверка Зеркального OUT

В PowerShell удобно использовать here-string:

```
@'
import sys
from pathlib import Path
sys.path.insert(0, str(Path.cwd()))
from tempfile import TemporaryDirectory
from system_core.services.media_service import _output_path
from Scripts.Common.script_runner import Runner

with TemporaryDirectory() as tmp:
    root = Path(tmp)
    source_root = root / 'Source'
    nested = source_root / 'Day1' / 'SceneA'
    nested.mkdir(parents=True)
    source = nested / 'clip.mov'
    source.write_text('x')
    target = _output_path(source, root / 'Out', 'wav_24_48', 'wav',
source_root=source_root, operation='Audio')
    expected = root / 'Out' / 'Audio' / 'wav_24_48' / 'Day1' / 'SceneA' / 'clip.wav'
    assert target == expected, (target, expected)

    runner = Runner('ff-x264-crf14', [])
    runner.source_dir = source_root
    runner.output_dir = root / 'OutCli'
    cli_target = runner.output_path(source, 'x264', 'mp4')
    expected_cli = root / 'OutCli' / 'Day1' / 'SceneA' / 'clip_x264.mp4'
    assert cli_target == expected_cli, (cli_target, expected_cli)
print('OK mirrored output paths')
'@ | runtime\python.exe -
```

Диагностика В GUI

Запустить:

1. Inventory.
2. Selftest.
3. Profile Doctor.

4. Hardware Capabilities.

5. Preset Test Matrix.

Ожидаемые **MISS** для отсутствующего GPU/backend не считаются ошибкой проекта.

Audio Smoke

На коротком тестовом файле:

1. **Аудиопотоки** ;
2. профиль **WAV 24-bit 48 kHz** ;
3. **Тест: первый файл** ;
4. **Команда** ;
5. реальный запуск;
6. проверить **audio_report.md/json** ;
7. проверить, что OUT сохраняет подпапку Source.

Remux Smoke

1. выбрать безопасную пару, например MP4 -> MKV или MKV -> MP4;
2. включить **Тест: первый файл** ;
3. dry-run;
4. запуск;
5. проверить report и output container.

LUT Smoke

1. положить **.cube** в **LUTs** ;
2. выбрать LUT явно;
3. выбрать **LUT x264** ;
4. проверить 18% gray preview, если профиль его показывает;
5. **Тест: первый файл** ;
6. dry-run;
7. запуск.

YouTube Smoke

Без реальной загрузки:

1. вставить тестовый URL;
2. выбрать профиль;
3. нажать `Команда` ;
4. проверить, что yt-dlp command содержит ожидаемый format/client/options.

Реальную загрузку делать только на маленькой ссылке или в отдельной test папке.

Документация

После changes в manifest/UI проверить ссылки:

- `USER_GUIDE_RU.md` ;
- `USER_GUIDE_EN.md` ;
- `Docs\README_RU.md` ;
- `README.md` ;
- relevant `Docs\docs\*.md` .

Новый режим или профиль должен быть описан хотя бы в одном user-facing guide.