Audion Media Tools and an AI agent

Русский · User guide · Console command · Operations

Contents

- Who does what
- The console command
  - Which operation
- Typical tasks
- Checking the result
- Rules

A guide for an AI agent (Claude Code, Codex and the like) a user has asked to work with the program. For a person: `USER_GUIDE_EN.md`; for the command line: `CLI_EN.md`.

Who does what

The program does the media work by FFmpeg, the same way at every run: encodes for an archive, for editing, for storage and for delivery; repackages into another container without re-encoding; extracts, puts back and converts sound; cuts a camera file without loss; changes the frame rate; applies a LUT; downloads by a link with yt-dlp; tells what the files are with ffprobe. Every button of its window is an operation of the console command `media_cli.cmd`, so an agent can do all of it.

The agent turns the user's words into an operation and its preset, runs it, reads the outcome and checks the file. The agent does not write FFmpeg commands of its own where an operation exists: the operation already knows the pitfalls of the containers, the frame rates and the timecode.

The console command

`media_cli.cmd` lies in the root of the program:

```
media_cli.cmd describe                                    operations, fields,
choices, presets (JSON)
media_cli.cmd run --operation ID --set NAME=VALUE         run an operation; --dry-run
- the commands it would start
media_cli.cmd run ... --source FILE_OR_FOLDER --target FOLDER --result OUTCOME.json
```

The Source is `input\` of the program unless `--source` names a folder or one file; the Target is `output\` unless `--target` names another. Exit codes: 0 done, 1 failed, 2 a wrong call or a run refused by its checks (nothing was run), 130 stopped. The outcome file of `--result`: `outcome`, `ok`, `message`, `parameters` - every value the run went with - and `data`: a media operation `processed` (files written), `skipped` (passed by), `files`, `failed` where it goes on past a failed file; a download `processed` and `downloaded` (the paths written); always `log_file`. All the keys: `CLI_EN.md`. `--result` is a `.json` file outside the Source; give it a full path, or call the command from the program folder: a relative path counts from the folder of the caller. `failed` or `refused` above 0 makes the outcome `failed`.

Which operation

Section	Mode	Call
AUDIO	EXTRACT	`media_cli.cmd run --operation audio_run --set audio_workflow=extract`
AUDIO	SOUND IN VIDEO	`media_cli.cmd run --operation audio_run --set audio_workflow=in_video`
AUDIO	CONVERT	`media_cli.cmd run --operation audio_run --set audio_workflow=audio_files`
ENCODE	ARCHIVE	`media_cli.cmd run --operation archive_run`
ENCODE	EDITING	`media_cli.cmd run --operation editing_run`
ENCODE	STORAGE	`media_cli.cmd run --operation storage_run`
ENCODE	DELIVERY	`media_cli.cmd run --operation delivery_run`
ENCODE	RAW	`media_cli.cmd run --operation raw_develop_run`
PACKAGE	VIDEO	`media_cli.cmd run --operation remux_run --set remux_stream_mode=video`

Section	Mode	Call
PACKAGE	AUDIO	`media_cli.cmd run --operation remux_run --set remux_stream_mode=audio`
PACKAGE	VIDEO AND AUDIO	`media_cli.cmd run --operation remux_run --set remux_stream_mode=video_audio`
PACKAGE	SUBTITLES	`media_cli.cmd run --operation subtitles_extract`
CUT	KEEP	`media_cli.cmd run --operation trim_run --set trim_action=keep`
CUT	CUT OUT	`media_cli.cmd run --operation trim_run --set trim_action=cut`
CUT	SPLIT	`media_cli.cmd run --operation trim_run --set trim_action=split`
FRAME RATE	VARISPEED	`media_cli.cmd run --operation fps_run --set fps_mode=varispeed`
FRAME RATE	CONFORM	`media_cli.cmd run --operation fps_run --set fps_mode=conform`
COLOR	LUT	`media_cli.cmd run --operation grading_run`
DOWNLOAD	BEST MP4	`media_cli.cmd run --operation youtube_run --set youtube_profile=best_mp4`
DOWNLOAD	SAFE MP4	`media_cli.cmd run --operation youtube_run --set youtube_profile=safe_mp4`
DOWNLOAD	OWN VIDEO	`media_cli.cmd run --operation youtube_run --set youtube_profile=video_custom`
DOWNLOAD	FOR UPLOAD	`media_cli.cmd run --operation youtube_run --set youtube_profile=upload_ready`
DOWNLOAD	OWN AUDIO	`media_cli.cmd run --operation youtube_run --set youtube_profile=audio_custom`
DOWNLOAD	SUBTITLES	`media_cli.cmd run --operation youtube_run --set youtube_profile=subs_only`

SOURCE INFO, SERVICE and the buttons beside a field:

Operation	What it is
`probe_source`	Probe Source: Run ffprobe once for Source, print a compact table and update the media status line.
`source_info`	Source Info: Run ffprobe for Source and print every audio stream: codec, sample format, sample rate, bit depth, channels, bitrate and tags.
`inventory`	Inventory: Count project folders.
`selftest`	Selftest: Check ffmpeg, ffprobe, yt-dlp and Deno.
`profile_doctor`	Profile Doctor: Validate GUI operations, script wrappers and the shared preset catalog.
`hardware_capabilities`	Hardware capabilities: Run short FFmpeg smoke tests for CPU, QSV/NVENC/AMF H.264 and HEVC, SVT-AV1 and dav1d.
`preset_test_matrix`	Preset Test Matrix: Run portable 1-second smoke tests across audio, remux, FPS, LUT, archive, editing, storage and delivery presets.
`install_7zip`	Install 7-Zip: Install or refresh portable 7-Zip; required before FFmpeg 7z payloads.
`install_ffmpeg`	Install FFmpeg: Install or refresh the portable FFmpeg of Gyan (GPL) in Tools\ffmpeg by the NVIDIA driver: the newest stable, 9.0.2 at least, without an NVIDIA card and from the LTS driver 610.88; below it 8.0.1 (570-610.87) or 7.1 so that NVENC works, and not every operation is possible.
`install_ytdlp`	Install yt-dlp: Install or refresh portable yt-dlp and Deno into Tools.
`install_mpv`	Install mpv: Install or refresh portable mpv into Tools/mpv: the player of the trim.
`install_cineraw`	Install ProRes RAW development: Build cineraw of libcraw into Tools/cineraw from its sources: a portable MSYS2 for the build (about 185 MB downloaded, 5 minutes), removed afterwards; cineraw develops ProRes RAW on a video card with Vulkan 1.4 (NVIDIA RTX).
`youtube_formats`	Formats of the link: What formats the link offers - resolution, codecs, size - listed in the journal, nothing downloaded. One of them is taken with the profile Own video.

Operation	What it is
`youtube_cookies_import`	Take under DPAPI: The cookies file of the field Cookies file sealed with DPAPI for this account of Windows into config\cookies.dpapi: downloads with Cookies - File take this copy, and the open file is no longer needed.

Every field, choice and preset of every operation: `OPERATIONS_EN.md`.

Typical tasks

1. **"Transcode the file for editing."** `probe_source` with `--source FILE` first: codec, size, frame rate, sound (`source_info` tells every sound stream in detail). Then `editing_run`: `editing_profile=prores_422_mov` by default, `prores_lt_mov` or `prores_proxy_mov` for lighter files, `dnxhr_hq_mov` where the editor wants DNxHR, an `*_mxf` preset for Avid. The file lands in a folder of the operation and of its settings - `output\Editing\prores422\` - not of the preset's name; `--dry-run` prints the exact path.
2. **"Make it smaller to keep" / "for upload".** `storage_run` (`x264_crf14`, `hevc10_crf14`, `svt_av1_crf14`, `x264_1080_crf14` to bring 4K down to 1080) or `delivery_run` (`review_h264`, `upload_h264`, `upload_hevc10`). A GPU encodes with `--set encode_backend=cuda` (NVIDIA), `amd`, `qsv`; `hardware_capabilities` tells what this machine can.
3. **"Put it into MP4 / MOV / MKV" without re-encoding.** `remux_run` with `remux_target_format` and `remux_stream_mode=video_audio`. A container that cannot hold a stream is refused before the run, with the reason.
4. **Sound.** `audio_run`: `audio_workflow=extract` takes it out of a video, `in_video` converts the sound inside a video with the picture copied, `audio_files` converts sound files; `audio_profile` - `wav_24_48`, `flac_24_48`, `m4a384`, `mp3_v0`, `lufs14_m4a` (loudness -14 LUFS), `lufs_report` (measures only).
5. **Cut.** `trim_run --set trim_file=NAME --set trim_action=keep --set trim_start=00:00:05,000 --set trim_end=00:01:00,000`. The video is copied: the start lands on the nearest keyframe, the end on the exact frame; the journal says where the cut really fell. `cut` takes a piece out, `split` cuts into parts at the points.
6. **Frame rate to 23.976.** `fps_run`: `fps_mode=varispeed` slows every frame and the sound with them, the pitch moves; `conform` keeps every frame too and stretches the sound, the pitch is kept.
7. **LUT, HDR to SDR.** `grading_run --set lut_file=LUTs\NAME.cube`; the LUTs live in `LUTs\`. `hdr_hable_x264` maps HDR to SDR without a LUT.

8. **Download by a link.** `youtube_formats --set youtube_url=LINK` shows what the link offers and downloads nothing; then `youtube_run --set youtube_url=LINK` (`youtube_profile=best_mp4` by default; `safe_mp4` for a link that breaks off; `audio_custom` with `youtube_audio_format=m4a` for the sound alone). `best_mp4` takes the best picture that fits MP4, which may be AV1; H.264 with AAC is `upload_ready`. `data.downloaded` names the file: `output\Download\<uploader>\<date> - <title> [<id>].<ext>`. A whole playlist: `--set youtube_source_mode=playlist`; a list of links: `batch` with `input\urls.txt`.

Checking the result

- Read the outcome file: `outcome` is `done` and `data.processed` is above 0. With `failed` or `refused` the journal (`data.log_file`) says why: its `[ERR]` and `[PREFLIGHT]` lines.
- A file already in the Target is not written over without `--set overwrite=true`: the journal says `[SKIP] NAME already exists and Overwrite is off`, the old file stays, and the outcome counts it in `skipped`, not in `processed`.
- Check the new file with `Tools\ffmpeg\bin\ffprobe.exe -v error -show_streams -show_format FILE`: codec, size, frame rate, duration and the sound against the source; for a cut - the number of frames.

Rules

- Do not change, move or delete the user's files in `input\` or wherever `--source` points. Results go to the Target.
- Do not read or print a cookies file. By default the user's cookies file goes with a download when there is one, as in the window; a public link needs none - `--set cookies_source=none`; a link that needs a sign-in takes `--set cookies_source=chrome` (or `edge`, `firefox`, `brave`).
- Clearing the folders deletes for good and is not run from the command: the user clears them in the window, through the Recycle Bin.
- Name a GPU only after `hardware_capabilities` said it works on this machine; the CPU always works.
- Before a long run, `--dry-run` shows the exact commands.