

Console command

[Русский](#) · [User guide](#) · [Operations](#) · [Wrappers](#) · [AI agent](#)

Contents

- [Commands](#)
- [Running an operation](#)
- [The output](#)
- [Outcome and exit codes](#)
- [Where the result goes](#)
- [Wrappers](#)
- [Clearing the folders](#)

`media_cli.cmd` in the root of the program runs what the buttons of the window run: an operation, with the same fields and the same presets. The window is the face of this command: whatever it starts, it starts through the command, and the first line of every run in its journal is a ready command that can be repeated in a console.

The command is for the cases where an operation is started not by a person at the window: a batch file, another tool, an AI agent, a long night of encodes.

Commands

Command	What it does
<code>describe</code>	The operations, their fields with choices and defaults, the presets and the layout of the window - one line of JSON.
<code>run</code>	Run an operation.
<code>wrapper NAME</code>	Run a wrapper of <code>Scripts\</code> by its name: <code>media_cli.cmd wrapper ff-x264-crf14</code> .

Command	What it does
<code>map</code>	The map of the program from its data: sections, operations, fields, presets, wrappers, the texts in both languages. <code>--out FILE</code> writes it to a file; <code>--docs</code> writes the reference documents (<code>OPERATIONS_*</code> , <code>SCRIPTS_*</code> and the tables of this guide).
<code>components</code>	The versions of yt-dlp, Deno, mpv, FFmpeg, 7-Zip and the GPU driver; <code>--latest</code> adds the newest releases (asked of GitHub at most once a day).
<code>plan --operation install_ffmpeg</code>	What an installer would download: the files, their source and size. Nothing is downloaded.
<code>version</code>	The version of the program.

`--root FOLDER` before the command names another copy of the program; `--language ru` or `en` - the language of the journal (the language of the window when not named).

Running an operation

```
.\\media_cli.cmd run --operation storage_run
.\\media_cli.cmd run --operation editing_run --set editing_profile=prores_lt_mov
.\\media_cli.cmd run --operation storage_run --set storage_profile=hevc10_crf14 --set
encode_backend=cuda `
    --source "D:\\Footage\\Day 1" --target "D:\\Encoded" --result
D:\\Encoded\\outcome.json
.\\media_cli.cmd run --operation youtube_run --set
"youtube_url=https://www.youtube.com/watch?v=..."
```

Key	Purpose
<code>-- operation ID</code>	The id of the operation. The table below gives the id of every mode of the window; <code>describe</code> gives all of them.
<code>--set NAME=VALUE</code>	The value of one field; may be repeated. A flag is <code>true</code> or <code>false</code> , a row of ticks - its values separated by commas (<code>--set target_codecs=x264,hevc_x265</code>). A preset field (<code>storage_profile</code> , <code>editing_profile</code> ...) sets the fields it names when it is named, and the other <code>--set</code> are laid over it; left unnamed, every field keeps its own default - the preset a form shows first is not applied by itself.

Key	Purpose
<code>--params</code> <code>FILE</code>	A JSON file "field: value". <code>--set</code> is laid over it.
<code>--source</code> <code>PATH</code>	A folder (its subfolders too) or one file. Without the key - <code>input</code> of the program.
<code>--target</code> <code>PATH</code>	A folder. Without the key - <code>output</code> of the program.
<code>--only</code> <code>KINDS</code>	The kinds of file to take from the Source: <code>mkv</code> or <code>mov,mxf</code> .
<code>--result</code> <code>FILE</code>	Where to write the outcome: a <code>.json</code> file outside the Source and outside <code>config\</code> . A full path, or call the command from the program folder.
<code>--cancel-</code> <code>file FILE</code>	The stop file: when it appears, FFmpeg is ended and a half-written file is taken away.
<code>--dry-run</code>	Run nothing: say the FFmpeg or yt-dlp commands the operation would start. An operation that cannot only say them is refused.

The names of the fields and their choices are those of `describe` and of [the list of operations](#). A field that is not set goes with its default - as in a form of the window just opened. A relative `--source` or `--target` is counted from the folder of the program; a relative `--result`, `--params` or `--cancel-file` - from the folder the command is called from.

What each mode of the window is

Section	Mode	Call
AUDIO	EXTRACT	<code>media_cli.cmd run --operation audio_run --set audio_workflow=extract</code>
AUDIO	SOUND IN VIDEO	<code>media_cli.cmd run --operation audio_run --set audio_workflow=in_video</code>
AUDIO	CONVERT	<code>media_cli.cmd run --operation audio_run --set audio_workflow=audio_files</code>
ENCODE	ARCHIVE	<code>media_cli.cmd run --operation archive_run</code>

Section	Mode	Call
ENCODE	EDITING	<code>media_cli.cmd run --operation editing_run</code>
ENCODE	STORAGE	<code>media_cli.cmd run --operation storage_run</code>
ENCODE	DELIVERY	<code>media_cli.cmd run --operation delivery_run</code>
PACKAGE	VIDEO	<code>media_cli.cmd run --operation remux_run --set remux_stream_mode=video</code>
PACKAGE	AUDIO	<code>media_cli.cmd run --operation remux_run --set remux_stream_mode=audio</code>
PACKAGE	VIDEO AND AUDIO	<code>media_cli.cmd run --operation remux_run --set remux_stream_mode=video_audio</code>
PACKAGE	SUBTITLES	<code>media_cli.cmd run --operation subtitles_extract</code>
CUT	KEEP	<code>media_cli.cmd run --operation trim_run --set trim_action=keep</code>
CUT	CUT OUT	<code>media_cli.cmd run --operation trim_run --set trim_action=cut</code>
CUT	SPLIT	<code>media_cli.cmd run --operation trim_run --set trim_action=split</code>
FRAME RATE	VARISPEED	<code>media_cli.cmd run --operation fps_run --set fps_mode=varispeed</code>
FRAME RATE	CONFORM	<code>media_cli.cmd run --operation fps_run --set fps_mode=conform</code>
COLOR	LUT	<code>media_cli.cmd run --operation grading_run</code>
DOWNLOAD	BEST MP4	<code>media_cli.cmd run --operation youtube_run --set youtube_profile=best_mp4</code>
DOWNLOAD	SAFE MP4	<code>media_cli.cmd run --operation youtube_run --set youtube_profile=safe_mp4</code>
DOWNLOAD	OWN VIDEO	<code>media_cli.cmd run --operation youtube_run --set youtube_profile=video_custom</code>
DOWNLOAD	FOR UPLOAD	<code>media_cli.cmd run --operation youtube_run --set youtube_profile=upload_ready</code>

Section	Mode	Call
DOWNLOAD	OWN AUDIO	<code>media_cli.cmd run --operation youtube_run --set youtube_profile=audio_custom</code>
DOWNLOAD	SUBTITLES	<code>media_cli.cmd run --operation youtube_run --set youtube_profile=subs_only</code>

SOURCE INFO, SERVICE and the buttons beside a field:

Operation	What it is
<code>probe_source</code>	Probe Source: Run ffprobe once for Source, print a compact table and update the media status line.
<code>source_info</code>	Source Info: Run ffprobe for Source and print every audio stream: codec, sample format, sample rate, bit depth, channels, bitrate and tags.
<code>inventory</code>	Inventory: Count project folders.
<code>selftest</code>	Selftest: Check ffmpeg, ffprobe, yt-dlp and Deno.
<code>profile_doctor</code>	Profile Doctor: Validate GUI operations, script wrappers and the shared preset catalog.
<code>hardware_capabilities</code>	Hardware capabilities: Run short FFmpeg smoke tests for CPU, QSV/NVENC/AMF H.264 and HEVC, SVT-AV1 and dav1d.
<code>preset_test_matrix</code>	Preset Test Matrix: Run portable 1-second smoke tests across audio, remux, FPS, LUT, archive, editing, storage and delivery presets.
<code>install_7zip</code>	Install 7-Zip: Install or refresh portable 7-Zip; required before FFmpeg 7z payloads.
<code>install_ffmpeg</code>	Install FFmpeg: Install or refresh portable FFmpeg Gyan GPL payload into Tools/ffmpeg.
<code>install_ytdlp</code>	Install yt-dlp: Install or refresh portable yt-dlp and Deno into Tools.
<code>install_mpv</code>	Install mpv: Install or refresh portable mpv into Tools/mpv: the player of the trim.
<code>youtube_formats</code>	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
<code>youtube_cookies_import</code>	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.

The output

The lines of the operation's journal, as in the window: `[OK]`, `[WARN]`, `[ERR]`, `[CMD]` - the command FFmpeg was started with, `[PREFLIGHT]` - the checks before the run, `[PROGRESS]` `clip.mp4 -> x264 [#####--] 42% 00:01:12/00:02:50 speed=3.4x` with FFmpeg's own line `frame= ... fps= ... time= ... speed=` under it, and FFmpeg's last word `Lsize=` at the end. The same lines go to the journal file in `logs\`, and the report of the run - to `report\`.

A program that reads the output by a pipe also gets service lines: `::media::{"started": ...}`, `{"progress": 0.42}`, `{"activity": ...}`, `{"finished": {"outcome": ...}}`. A console does not show them.

Outcome and exit codes

Code	Outcome	What it means
0	<code>done</code>	The operation finished.
1	<code>failed</code>	It ran and something broke, or a file failed or was refused: the journal says what.
2	<code>- /</code> <code>refused</code>	A wrong call - a field the operation has not, a value that is none of its choices, <code>--dry-run</code> of an operation that cannot only say its commands; or the checks before the run refused it - no file of its kind in the Source, no FFmpeg. Nothing was run.
130	<code>stopped</code>	Stopped by the stop file (<code>--cancel-file</code>), before the first file or during one.

The file of `--result`:

```
{
  "operation": "storage_run",
  "outcome": "done",
  "ok": true,
  "message": "Operation finished.",
  "data": {
    "processed": 1,
    "files": 1,
    "log_file": "...\\logs\\2026-10-15 22-34-54-804_storage_run.log",
    "batch_report_md": "...\\report\\2026-10-15 22-34-54-804_storage_run\\batch_report.md"
  },
  "parameters": { "storage_profile": "x264_crf14", "target_codecs": ["x264"], "crf": 14, "...": "..." },
  "source": "D:\\Footage\\Day 1",
  "target": "D:\\Encoded",
  "seconds": 2.1
}
```

parameters are all the values the operation ran with, the defaults included: the run can be repeated from them exactly. **data** is what the operation tells: a media operation **processed** (files written), **skipped** (passed by, the reason in the journal), **files** (files of the Source), **failed** where it goes on past a failed file and **refused** where it turned a file down; a download **processed**, **downloaded** - the paths of what it wrote - and **existing**, found there already.

Where the result goes

Into a folder of the operation in the Target, and inside it a folder of the settings - named by the target and its values, not by the preset: **Storage\<x264_crf14_slow\clip.mp4**, **Editing\prores422\clip.mov**, **Archive\...**, **Delivery\...**, **Audio\...**, **Remux\...**, **Trim\...**, **FPS\...**, **Color\...**, **Subtitles\...**; the subfolders of the Source are kept below it. **--dry-run** prints the exact path. Downloads go to **Download\<uploader>\<date> - <title> [<id>].<ext>** (a playlist: **Download\<number> - <title> [<id>].<ext>**).

A file already there is not written over unless **--set overwrite=true**: the journal says **[SKIP]** **NAME already exists and Overwrite is off**, FFmpeg is not started for it, and the outcome counts it in **skipped**, not in **processed**.

Wrappers

95 files of `Scripts\` - `ff-*`, `audio-*`, `yt-*` - each a ready task with its values, called by a double click, from a console or by name through the command: `media_cli.cmd wrapper ff-prores-lt-mov`. Each is listed with the same call through `run` in [the list of wrappers](#). The wrappers of a trim take the file and the points from the variables `AUDION_TRIM_FILE`, `AUDION_TRIM_START`, `AUDION_TRIM_END`; the downloads - the links after the name, or `input\urls.txt`.

The menus `launcher_project.cmd`, `launcher_tools.cmd` and `cli\launcher-*.cmd` find a wrapper by typing (FZF).

Clearing the folders

The operations that clear the Source, the Target or the working folder delete for good and are not run from the command: the window clears them through the Recycle Bin.