

Audion Media Tools — User Guide

[Русский](#) · [About](#) · [Console command](#) · [Operations](#) · [Measurements](#)

Contents

- [The Window](#)
- [The Order of Work](#)
- [The Source and the Target](#)
- [Cutting: the Player and Its Keys](#)
- [The Sections](#)
- [Hardware](#)
- [The Journal and Reports](#)
- [SETTINGS and ABOUT](#)
- [Rules for Safe Work](#)
- [Technical Reference](#)

How to work with it: the window, the order of running an operation, the Source and the Target, cutting with the player, the journal and the reports.

The Window

- **The header:** the name and version; in the middle the versions of what the program runs — yt-dlp, Deno, mpv, FFmpeg and the video card driver; on the right SERVICE, the language (RU / EN), the light or dark look, SETTINGS and ABOUT.
- **The ribbon of sections:** AUDIO, ENCODE, PACKAGE, CUT, FRAME RATE, COLOR, DOWNLOAD — under each name a line of what it does.
- **The form** on the left: the modes of the section, then the blocks of fields — the preset first, then what it can be changed with. Above the form stand RUN, ↶ (back to the preset of the mode) and COMMAND.
- **The Source and the Target** on the right, with the buttons of the folders.

- **The journal** under them: what the run says, in the colours of the program that says it, with FFmpeg's own progress line.
- **COMMAND** at the bottom: a line for a command of your own.

Every section and mode keeps its own form, and the form is remembered between launches. Only the encoder and the decoder are shared by all: they are a choice of the machine, not of the task.

The Order of Work

1. Choose the Source and the Target — or leave **input** and **output**.
2. Open the section and the mode.
3. Pick a preset.
4. Change only what needs changing.
5. Press COMMAND — the journal shows exactly what will go to FFmpeg, and nothing is run.
6. Run the first file.
7. Run the folder.

The preset sets a ready base; the fields below override it selectively. A tick never enables a hidden action the preset did not set.

Steps 5 and 6 are not skipped on a new operation. A wrong parameter costs seconds on one file and an hour on a folder.

The Source and the Target

button	what it does
folder	choose the folder with the originals (or the folder for the result)
file	one file as the Source: the operation goes through it alone
history	paths used before, the pinned ones first
pin	keep this path at the top of the history
open	open the folder in Explorer; one file is shown selected
RESET	both paths back to the folders of the program, input and output ; files are not touched
LIST	the names of the files of the Source in the journal, with their subfolders

button	what it does
DETAILS	what ffprobe knows of the Source: a table of the files, or every sound stream
TO SOURCE	the result becomes the Source of the next operation: the Source is cleared into the Recycle Bin, then the Target is moved into it
CLEAR	the contents of the Source, the Target or both — into the Recycle Bin; the folders stay, and the window asks first

A path may also be typed or dropped on the window. An empty field means the folder of the program.

The Source is walked recursively and the output mirrors its structure:

Target\<operation>\<preset>\<as in the Source>\<file name>.<ext>

`input\Day1\clip.mov` and `input\Day2\clip.mov` do not collapse into one flat folder and fight over a name.

Cutting: the Player and Its Keys

Cut points are found in the mpv player — `Open in the player` in the CUT section — and carried into the fields by `Take A and B`, or one at a time by `Mark IN` and `Mark OUT`. The keys work **in the player window**, not in the program.

The file is chosen by the arrows beside its name; the button beside them tells what ffprobe knows of it. One file is cut at a time: the takes of a folder do not share cut points.

The player writes into the last row. `+` after it adds the next row once the last one holds its points, and clears A and B in the player so the next piece is looked for from scratch. For KEEP and CUT OUT each row is a variant of its own, written to its own file — `_keep01`, `_cut02` — and rows may overlap; beside OUT stands the length of the piece. For SPLIT each point has one field, the buttons become `Take A` and `Mark here`, and a point is taken only past the ones before it. Points can always be typed as well: digits one after another from the hours, the separators come by themselves and what is left unfilled becomes zeros — `000128` becomes `00:01:28,000`; plain seconds work too. `▲▼` in a field step the section the caret stands in — hours, minutes, seconds, or a frame (a millisecond for sound); without a caret, the seconds; held, the step speeds up. Points are taken only while the player shows the file chosen in the section; after the arrows, press `Open in the player`.

The dozen keys that do the whole job:

key	what it does
Space	play or pause
, .	step exactly one frame back or forward
← →	seek 5 seconds
Shift+← Shift+→	exactly one second back or forward
Shift+↓ Shift+↑	exactly five seconds
↓ ↑	one minute back or forward
Home	to the start of the file
l	set point A ; press again for B ; the piece between them loops
Ctrl+l	clear A and B
b	play backwards
n	return to playing forwards
[]	10% slower or faster — at 0.25 every syllable is audible
Backspace	back to normal speed
s	a frame grab next to the source file

l is the cut: set A and B, listen to the loop, and if a point is wrong nudge it with **,** and **.** and press **l** again.

The difference between ← and Shift+← matters. An ordinary seek jumps five seconds and shows the nearest frame; **Shift** is exact, precisely one second, and lands on the frame. Finding a cut point needs the second one.

b and **n** are this program's own: in stock mpv **b** is debanding, overridden here (`config\mpv\input.conf`).

The full list — all 197 bindings with their player commands — is in `tools\MPV_HOTKEYS_EN.md`. It was taken from the player itself through `input-bindings`, not copied from a web page.

Cutting in detail — the working cycle, the three actions, accuracy, refusals — is in `tools\TRIM_EN.md`; what is compatible with what when cutting is in `tools\TRIM_MATRIX_EN.md`.

The Sections

section	about	in detail
AUDIO	extraction, resampling, loudness, packaging; sound inside a video without re-encoding the picture	<code>tools\AUDIO_WORKFLOWS_EN.md</code>
ENCODE	lossless FFV1 and x264, ProRes and DNxHR, x264/x265/AV1, upload profiles	<code>tools\ENCODING_PIPELINES_EN.md</code>
PACKAGE, FRAME RATE, COLOR	container changes, frame rate, lookup tables, HDR to SDR	<code>tools\REMUX_FPS_LUT_EN.md</code>
CUT	frame-accurate and keyframe cuts; sound without loss (WAV, FLAC, ALAC) on the sample, compressed sound on its packets without re- encoding, or on the sample with Exact	<code>tools\TRIM_EN.md</code>
DOWNLOAD	a link, a playlist or a list, format profiles, resolution, subtitles, cookies	<code>tools\YOUTUBE_DOWNLOAD_EN.md</code>
SERVICE	what hardware is present, what it can do, profile checks, installers	<code>tools\HARDWARE_DIAGNOSTICS_EN.md</code>

Every field, choice and preset with its tooltip: [Operations](#).

DOWNLOAD. `PASTE` beside the link, the list file and the cookies file takes them from the clipboard. ⓘ beside the link shows what formats it offers, nothing downloaded. A link that wants a sign-in takes the cookies of a browser, or of a file — the shield beside it seals that file with DPAPI for this Windows account, so the open file is no longer needed.

COLOR. The LUTs live in the `LUTs` folder of the program. The buttons beside the list add files (`.cube`, `.3dl`, `.lut`) or a whole folder — they are copied into `LUTs` — and open the folder.

Hardware

The program can choose between the NVIDIA, Intel, and AMD hardware paths, separate AV1 encoder and decoder, and the plain processor.

Run SERVICE → Hardware capabilities before the first run on a new machine. The result is cached and hardware profiles are checked against it. A missing path is a normal answer, not a fault: it means the driver or the hardware is absent.

The Journal and Reports

The journal shows the lines of the run as the programs write them: **[CMD]** — the command FFmpeg was started with, **[PROGRESS]** with FFmpeg's own line under it, **[OK]**, **[WARN]**, **[ERR]**. A run opens with its name and the console command that repeats it, and ends with its outcome and time. Under the journal stand the operation, its time and its outcome; STOP ends a run, and a half-written file is taken away.

Above the journal: LOGS — the journals of the runs, REPORT — their reports, CONFIG — the settings of the program, then copy the journal, clear it, and widen it (the width is remembered).

Every operation writes **batch_report.md** and **.json**. AUDIO adds **audio_report.md** and **.json**: the source stream, input parameters, processing mode, loudness, output path.

COMMAND at the bottom runs a command of your own — in the PowerShell 7 of the program (PS) or in **cmd.exe** (CMD). Its lines go to the journal, STOP ends it; while an operation runs, a command waits its turn. The buttons beside the line: the history (pinned commands first), the path of a file in quotes, the folder the command runs in.

SETTINGS and ABOUT

SETTINGS: the four looks (Audion Code Dark, Audion Vault Dark, Audion Dark, Audion Light), the size of the window's text and of the terminal's; COMPONENTS — the versions of yt-dlp, Deno, mpv, FFmpeg and 7-Zip with UPDATE or INSTALL, and CHECK NOW for the newest releases; SYSTEM — the processor, the video card and its driver, the memory, which FFmpeg coders passed the check, the disk, and COPY for a report of a trouble.

ABOUT: the version, the folders of the program, the author, and what the program is made of with the licence of each part.

Rules for Safe Work

- A new operation: COMMAND first, then the run.
- A large folder: the first file first.

- Do not enable overwriting unless you are sure of the target folder.
- A hardware path: check the hardware capabilities first.
- A colour table: look at which `.cube` is selected.
- A download batch: check the list of links first.
- Clearing and TO SOURCE: look carefully at the folders — the window names them before it asks.

Technical Reference

Running

`Start.exe` opens the window. If the Python environment or a tool is missing, `builder_main.cmd` builds it.

The same operations run from a console or a script: `media_cli.cmd` — [Console command](#).

Checks After Changes

```
runtime\python.exe -m pytest -q -p no:cacheprovider tests system_core\tests
install\Build-App.cmd
```

The first is the engine (about five minutes — the wrappers and the cuts run the real FFmpeg); the second checks the window, builds it into `App\` and lets the built copy check itself. What is run before a release: the [checklist](#) (Russian).

Tooltips

Every field, choice and preset has its tooltip in both languages — hold the cursor on it. The same texts stand in [Operations](#).