

# FilmCraft

ALPHA

clone of Adobe Premiere Pro

Edit, grade, mix, caption and export video by describing the cut. Claude drives FilmCraft through the `filmcraft` and `filmcraft-live` MCP servers and shows you the frame it just made.

## v0.8.0

Driven from Claude Code via MCP  
Richard Young • richard@deepneuro.ai  
2026-10-15

**Alpha software.** FilmCraft is a pre-1.0 ArtCraft app. Expect missing features, rough edges and the occasional crash; keep originals safe and let Claude work on copies.

# 17

MCP tools

# 701

engine commands

# 303

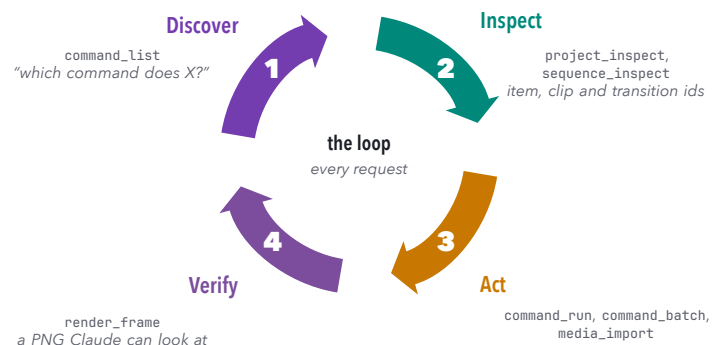
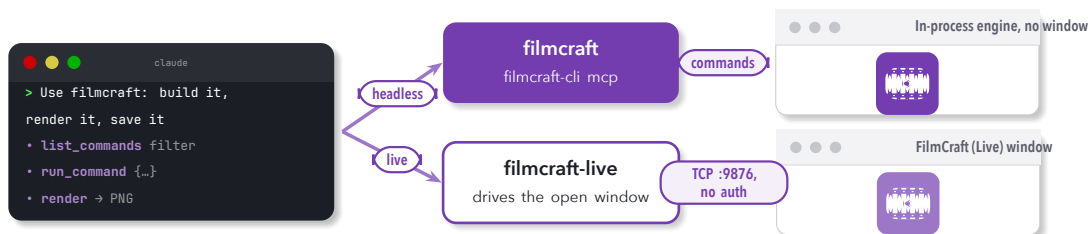
effects &amp; transitions

# 2

MCP servers

## 01 The idea in one picture

Every menu item, button and timeline gesture in FilmCraft is an **engine command** with a dotted id and typed parameters (`timeline.razor`, `effects.setParam`, `file.exportMedia`, ...). The MCP server is a thin layer over them. You describe the edit; Claude discovers the command, reads the ids it needs, runs it, and renders a frame so both of you can see the result.



**i Check it is wired up**

Type `/mcp` in Claude Code: `filmcraft` should be *connected* with 17 tools. `filmcraft-live` only connects while FilmCraft (Live) is running. The headless project lives in memory until Claude saves it (`file.saveAs`) or exports, so always finish with “save it to ...”.

**i Time is measured in ticks**

**254 016 000 000 ticks = 1 second**, which divides evenly by every common frame and sample rate (23.976 and 29.97 DF are exact). Commands also accept `seconds`, `frame` or `timecode`, so talk to Claude in seconds or timecode and let it convert.

## 02 First five minutes

Both frames on this page are real output from FilmCraft, captured through the MCP server while writing this guide. The footage was three plain PNG stills; swap in your own clips.

- 1 **Prove the connection** “Use filmcraft: list the commands that contain lumetri.” Claude calls `command_list` and shows ids with their parameter docs.
- 2 **Import and lay a cut** Point Claude at a folder of clips; it imports them, makes a sequence and places them on V1 (storyboard 1).
- 3 **Add a graphic** A lower third is one command, `graphics.newText`, once the parameters are right (storyboard 1 shows the mistake and the fix).
- 4 **Dissolve and grade** Transitions and Lumetri are commands too (storyboard 2). Always end with “render a frame at ...”.
- 5 **Save** “Save the project as `~/Videos/rough.fcproj` and export the YouTube 1080p preset.”

## Storyboard 1: import, sequence, lower third

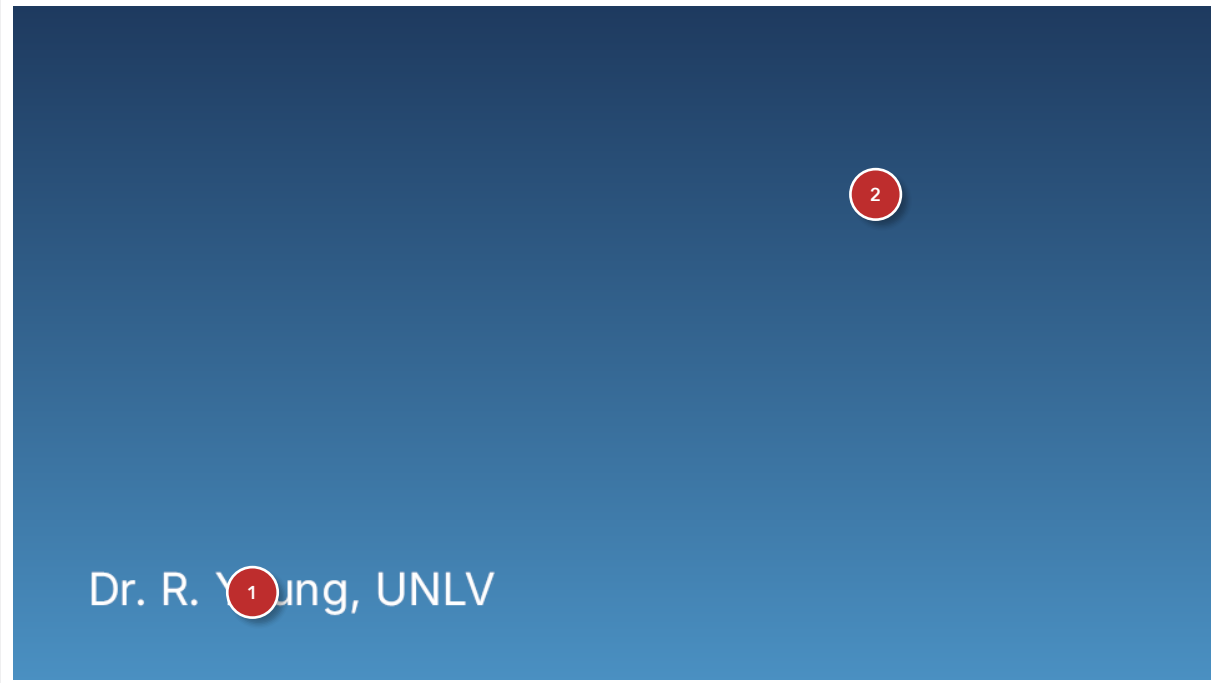
### YOU, IN CLAUDE CODE

"Use filmcraft. Import the three stills in ~/demo/stills, make a 1080p 23.976 sequence called Rough cut, lay them on V1, and put a lower third "Dr. R. Young, UNLV" over the first 5 seconds. Render the frame at 2 s."

### MCP CALLS CLAUDE MADE

- 1 **media\_import** {"paths": ["../shot-a.png", "shot-b.png", "shot-c.png"]} → items [1, 2, 3]
- 2 **command\_batch** file.newSequence {"name": "Rough cut", "width": 1920, "height": 1080, "fps": 23.976} → sequence 10; timeline.place {"items": [1,2,3], "track": "V1", "seconds": 0} → clips [11, 12, 13]; graphics.newText {..."seconds": 0}
- 3 **sequence\_inspect** the text clip is 1 frame long: seconds is the *duration*, *time* (ticks) is the start
- 4 **command\_run** graphics.newText {"text": "Dr. R. Young, UNLV", "position": [120, 960], "size": 72, "seconds": 5, "track": 2, "time": 0} → clip 18; then render\_frame {"seconds": 2}

### WHAT CAME BACK



- 1 The lower third from **graphics.newText**, placed at [120, 960], on V3 after V2 reported "not free here".
- 2 Shot A itself: a plain gradient still, 120 frames long, placed by **timeline.place**. Ids (1-3, 11-13, 18) all came from tool results, never guessed.

## Storyboard 2: a dissolve and a warm grade

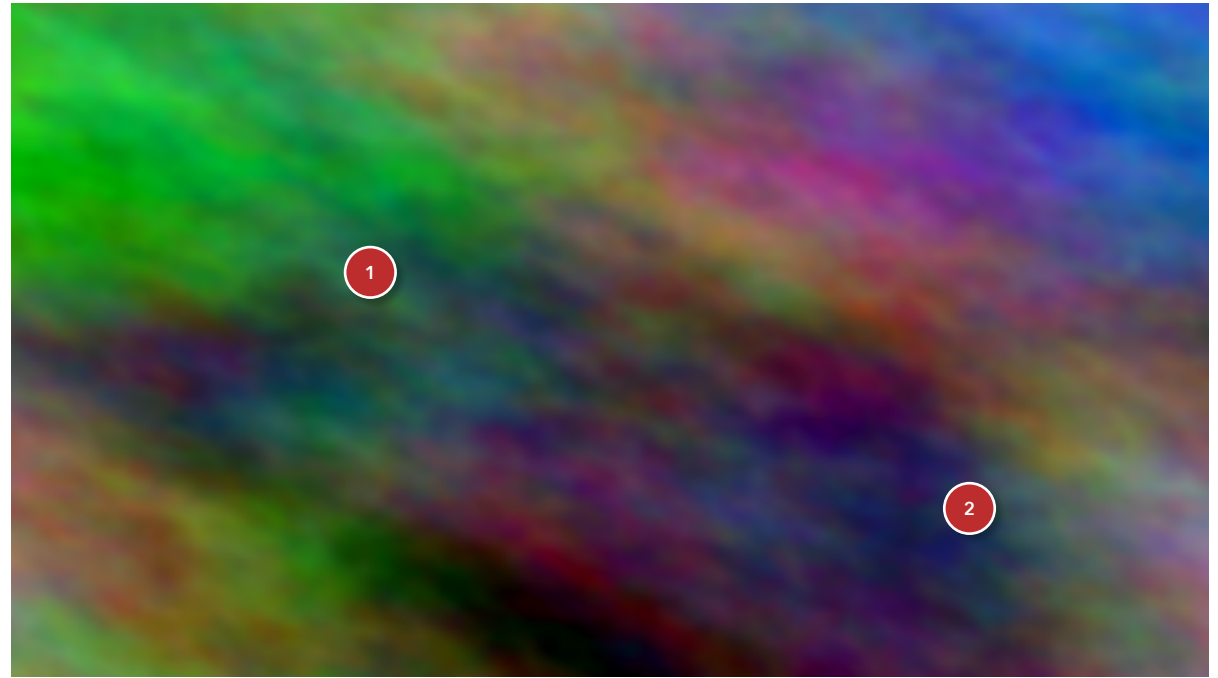
### YOU, IN CLAUDE CODE

"Put a 12-frame cross dissolve going into the second clip, apply Lumetri to the third clip and warm it up (temperature +30). Show me a frame at 11 s."

### MCP CALLS CLAUDE MADE

- 1 `command_batch` `sequence.applyVideoTransition`  
`{ "clip": 12, "effect": "cross_dissolve", "frames": 12,`  
`"edge": "in" } → transition 17`
- 2 `command_batch` `effects.apply` `{ "clip": 13, "effect":`  
`"lumetri"; effects.setParam { "clip": 13, "effect":`  
`"lumetri", "param": "temperature", "value": 30 }`
- 3 `render_frame` `{ "seconds": 11, "max_side": 960 }`

### WHAT CAME BACK



- 1 Shot C at 11 s through Lumetri with temperature +30: the plasma still shifts warmer.
- 2 `sequence_inspect` afterwards lists every Lumetri parameter on the clip (exposure, curves, wheels, vignette...), so "what did you change?" has a precise answer.

## Prompts to explore with



"Use filmcraft: list the commands that contain "lumetri" and show me their parameters."

`command_list` with a filter: how Claude finds anything.



"Inspect the active sequence and tell me every clip's start time in seconds."

`sequence_inspect` returns ticks; Claude divides by 254 016 000 000.



"Import everything in ~/Videos/shoot into a bin Raw, make a 4K 29.97 sequence and lay the clips in filename order."

Import, bin, sequence, place.

●●● warm-up ●●● real work ●●● stretch



"Razor V1 at 0:42 and 1:05, ripple-delete the middle and close gaps, then render frames either side of the cut."

Edit, then verify with two `render_frame` calls.



*"Apply the Rec.709 LUT for S-Log3 to clip 12, read the waveform, and tell me if anything clips."*

`lumetri.setInputLut` plus `scopes.read`.



*"Transcribe the sequence, cut filler words and pauses over 1.5 s, make two-line captions, and burn them into a YouTube export."*

Needs a Whisper model; Claude chains six commands.



*"Build a 60-second trailer from the markers in this project: 3-second slices at each marker, dissolves between, music ducked under the dialogue."*

Stretch: markers, place, transitions, Essential Sound.

## 03 Goals → how

Each card: a prompt to paste, the MCP tools Claude reaches for, and the command ids and parameters. Clip and item ids always come from an inspect call first; Claude does that bookkeeping.

### ■ Ingest & organise

#### 1 Start a project from a shoot

##### ASK CLAUDE

*"New project Interview; import every clip in ~/Videos/shoot into a bin called Raw; make a 1080p 23.976 sequence."*

TOOLS `command_run` → `media_import`  
→ `project_inspect`

KEY `file.newProject`, `file.newBin`, `media_import`  
`paths` (absolute) + `bin`, `file.newSequence` `width`  
`height` `fps`

#### 2 Proxies for heavy 4K footage

##### ASK CLAUDE

*"Make half-res H.264 proxies for everything in Raw and switch playback to proxies."*

TOOLS `project_inspect` → `command_run`

KEY `media.createProxies` `items` `preset` (e.g. `h264_half`, `prores_proxy_quarter`) `wait`;  
`media.toggleProxies`

## Assemble & cut

### 3 Rough cut from selects

#### ASK CLAUDE

"Lay clips 3, 7, 4 and 9 back to back on V1, with 12-frame cross dissolves between them."

TOOLS `command_batch` → `sequence_inspect`

KEY `timeline.place items track seconds;`  
`sequence.applyVideoTransition effect`  
`cross_dissolve frames`

● Or `clip.automateToSequence` (Premiere's Automate to Sequence).

### 6 Two-camera interview (multicam)

#### ASK CLAUDE

"Sync cam A, cam B and the lav by audio into a multicam clip; cut to camera 2 at 0:45, back to 1 at 1:10."

TOOLS `command_run` → `sequence_inspect`

KEY `clip.createMulticam method audio;`  
`multicam.switchAngle camera time;`  
`clip.multicamFlatten`

### 4 Cut a section and close the gap

#### ASK CLAUDE

"Razor V1 at 1:12 and 1:31, ripple-delete the middle piece, and close any gaps on V1."

TOOLS `command_run` → `sequence_inspect`

KEY `timeline.razor seconds track;`  
`edit.rippleDelete clips; sequence.closeGap`

● Fine trims: `timeline.trim`, `timeline.roll`,  
`timeline.slip`, `timeline.slide`.

### 5 Split a long reel into shots

#### ASK CLAUDE

"Detect scene changes in the archive reel, cut at each one and drop a marker on every cut."

TOOLS `command_run`

KEY `clip.sceneEditDetection sensitivity (0-100)`  
`minShotFrames applyCuts createSubclips`  
`generateMarkers`

## Look: colour, graphics & motion

### 7 Log footage to Rec.709, plus a look

#### ASK CLAUDE

"A-cam is S-Log3 / S-Gamut3.Cine: apply the matching Rec.709 LUT, then the Golden Dusk preset, and show me a frame."

TOOLS `command_run` → `render_frame`

KEY `lumetri.setInputLut lut`  
`builtin:slog3-sgamut3cine-to-rec709;`  
`lumetri.applyPreset name; own .cube:`  
`lut.import + lumetri.setLook`

### 8 Hand-grade and read the scopes

#### ASK CLAUDE

"Warm this shot a little, exposure +0.3, saturation 90, then check the waveform for clipping."

TOOLS `command_run` → `render_frame`

KEY `effects.apply lumetri; effects.setParam param`  
`(temperature, exposure, saturation, ...) value;`  
`scopes.read`

● `lumetri.applyMatch` matches a shot to a reference frame (storyboard 2 used temperature).

### 9 Titles and lower thirds

#### ASK CLAUDE

"Add a lower third 'Dr. R. Young, UNLV' at 0:04 for 5 s on V3; use a built-in template if one fits."

TOOLS `command_run` → `render_frame`

KEY `graphics.newText text position [x,y] size`  
`seconds (duration) time (start, ticks) track`  
`(0-based); graphics.template.list /`  
`graphics.template.apply values`

● Templates are `.fcgt`; Adobe `.mogrt` files are refused.

**10 Ken Burns push-in with easing****ASK CLAUDE**

"On the opening still, keyframe scale 100 to 115 across the clip and ease both keyframes."

**TOOLS** `sequence_inspect` → `command_batch`

**KEY** `effects.setParam effect motion param scale time keyframe; effects.setInterpolation (easeIn, easeOut, bezier, hold, ...)`

**11 Blur a moving face or plate****ASK CLAUDE**

"Blur the plate in the clip at 0:22 with an elliptical mask and track the mask forward."

**TOOLS** `command_run` → `render_frame`

**KEY** `effects.apply gaussian_blur; masks.add shape center size; masks.track direction; masks.set feather`

● Shaky shot instead? `effects.apply warp_stabilizer.`

**12 Slow motion and freeze frames****ASK CLAUDE**

"Play the clip at 2:05 at 40% speed with optical flow and ripple the rest of the timeline."

**TOOLS** `sequence_inspect` → `command_run`

**KEY** `clip.speedDuration speed (%) ripple interpolation opticalFlow; freezes: clip.frameHold`

**Audio****13 Clean dialogue, duck the music****ASK CLAUDE**

"Tag interview clips as dialogue with noise reduction, match them to -16 LUFS, and duck the music under the voice."

**TOOLS** `command_batch`

**KEY** `essentialSound.setType (dialogue|music|sfx|ambience); essentialSound.set key repair.noise.on; essentialSound.autoMatch target; essentialSound.generateDucking`

**14 Scratch narration from text****ASK CLAUDE**

"Read this paragraph as narration on A3 starting at 0:30, a little slower than default."

**TOOLS** `command_run`

**KEY** `tts.voices; tts.create text voice pace (0.5-2) time track`

● Natural voices: `tts.downloadVoices` (Kokoro-82M, about 335 MB).

**Captions & transcripts****15 Captions from an SRT, styled****ASK CLAUDE**

"Import ~/Videos/interview.srt as captions, 42-px white on a translucent box at the bottom."

**TOOLS** `command_run` → `render_frame`

**KEY** `captions.import path; captions.setStyle size color background anchor; captions.export (srt, vtt, scc, ttml, ...)`

**16 Transcribe and cut the ums****ASK CLAUDE**

"Transcribe the sequence, remove filler words and pauses over 1.5 s, then make two-line captions."

**TOOLS** `command_run` (several)

**KEY** `transcript.downloadModel; sequence.transcribe; transcript.removeFillers; transcript.removePauses minSeconds; transcript.createCaptions lines`

● Needs a Whisper model (tiny/base/small), downloaded on first use.

## Deliver & hand off

### 17 H.264 for YouTube

ASK CLAUDE

*"Export to ~/Videos/out/final.mp4 with the YouTube 1080p preset at -14 LUFS, captions burned in; wait for it."*

TOOLS `command_run`

KEY `file.exportMedia path preset loudnessLufs burnCaptions wait`; or `format h264 + crf`

● Presets: `export.presets.list` (YouTube, Vimeo, Social Vertical, ...).

### 18 Several deliverables in one queue

ASK CLAUDE

*"Queue a ProRes HQ master, the YouTube H.264 and a WAV of the mix, then run the queue."*

TOOLS `command_batch`

KEY `export.queue.add path + settings (format proresProfile preset); export.queue.start wait; jobs.list`

### 19 Hand off to Resolve, Avid, Pro Tools

ASK CLAUDE

*"Export FCP7 XML and OTIO for Resolve, an AAF with embedded audio for the mixer, and markers as CSV."*

TOOLS `command_batch`

KEY `file.exportInterchange format (edl|xml|fcpxml|otio|aaf|omf); file.exportAaf audio handles; markers.exportCsv`

### 20 Thumbnail still and save

ASK CLAUDE

*"Export the frame at 1:02 as a 16-bit PNG, then save the project as ~/Videos/interview.fcproj."*

TOOLS `command_run` → `command_run`

KEY `file.exportFrame path format depth time; file.saveAs path; snapshot: file.saveCopy`

### ✓ Always add: "render a frame and check"

`render_frame` returns a PNG of the Program frame at any `seconds`; `scopes.read` returns waveform and vectorscope numbers. Ask for both after a grade. If a command does not exist (there is no `edit.delete`, for example), ask Claude to "list commands matching delete" or to `edit.undo`.

## 04 The command map (701 ids behind `command_run`)

Every menu item, shortcut and timeline gesture is a dotted command id. Darker tiles are bigger families. Ask Claude to “list filmcraft commands matching *word*” to open any tile.

|                                                                                       |                                                                                               |                                                                                               |                                                                                                                             |                                                                                              |                                                                    |
|---------------------------------------------------------------------------------------|-----------------------------------------------------------------------------------------------|-----------------------------------------------------------------------------------------------|-----------------------------------------------------------------------------------------------------------------------------|----------------------------------------------------------------------------------------------|--------------------------------------------------------------------|
| <b>82</b> <b>graphics.</b><br>text, shapes, layers, styles, motion-graphics templates | <b>57</b> <b>file.</b><br>projects, sequences, import, save/recover, exportMedia, interchange | <b>57</b> <b>clip.</b><br>speed, frame holds, audio gain, scene detection, multicam, subclips | <b>56</b> <b>timeline.</b><br>place, move, trim, roll, slip, slide, razor, track settings                                   | <b>45</b> <b>sequence.</b><br>settings, transitions, render, add edit, close gap, transcribe | <b>40</b> <b>project.</b><br>bins, views, metadata, search         |
| <b>39</b> <b>edit.</b><br>undo, ripple delete, paste attributes, selection            | <b>35</b> <b>multicam.</b><br>angle switching, flatten, sync                                  | <b>32</b> <b>markers.</b><br>in/out, markers, chapters, CSV export                            | <b>24</b> <b>trim.</b><br>trim mode, dynamic trim                                                                           | <b>22</b> <b>captions.</b><br>caption tracks, styles, import/export                          | <b>20</b> <b>mixer.</b><br>track mixer, inserts, sends, automation |
| <b>19</b> <b>export.</b><br>presets, export queue                                     | <b>17</b> <b>media.</b><br>relink, proxies, find missing                                      | <b>16</b> <b>effects.</b><br>apply, parameters, keyframes, interpolation                      | <b>140</b> <b>other</b><br>shortcuts, playhead, transcript, masks, essentialSound, lumetri, lut, tts, presets, scopes, jobs |                                                                                              |                                                                    |

## 05 Tool reference (all 17 MCP tools)

| MCP tool                                  | What it does                                                                                                                      |
|-------------------------------------------|-----------------------------------------------------------------------------------------------------------------------------------|
| <b>Commands</b> - the path for every edit |                                                                                                                                   |
| <code>command_list</code>                 | Every command: id, label, menu path, shortcut, parameter doc, enabled-now. Args <code>filter</code> , <code>enabled_only</code> . |
| <code>command_run</code>                  | Run one command: <code>id</code> + <code>params</code> . Undoable edits land in History.                                          |
| <code>command_batch</code>                | Run <code>steps</code> in order; <code>stop_on_error</code> (default true). Each edit is its own undo step.                       |
| <b>Inspect, import, preview</b>           |                                                                                                                                   |
| <code>doc_inspect</code>                  | Whole project as JSON: tree plus active sequence (null if none).                                                                  |
| <code>project_inspect</code>              | Bins and items with ids, types, durations; active sequence; source clip.                                                          |
| <code>sequence_inspect</code>             | Settings, tracks, clips (ids, start/duration in ticks and frames, effects), transitions, markers, in/out, playhead, selection.    |
| <code>media_import</code>                 | Import by absolute <code>paths</code> ; optional <code>bin</code> , <code>image_sequence</code> .                                 |
| <code>render_frame</code>                 | PNG of the active sequence at <code>seconds</code> (default: playhead); <code>max_side</code> default 960.                        |

| MCP tool                                       | What it does (cont.)                                                                                                                                        |
|------------------------------------------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <code>render_preview</code>                    | Same as <code>render_frame</code> ; in bridge mode, a Program-monitor screenshot.                                                                           |
| <b>Live app only</b> ( <i>filmcraft-live</i> ) |                                                                                                                                                             |
| <code>ui_elements</code>                       | On-screen interactive elements (id, label, rect); <code>prefix</code> filter, e.g. <code>timeline.clip</code> .                                             |
| <code>ui_click</code>                          | Click by element <code>id</code> (e.g. <code>tools.Razor</code> ) or <code>x/y</code> ; <code>button</code> , <code>count</code> , <code>modifiers</code> . |
| <code>ui_drag</code>                           | Press-drag-release <code>from</code> → <code>to</code> : move clips, trim edges, scrub.                                                                     |
| <code>ui_key</code>                            | Press a key or shortcut: Space, Cmd+K, Shift+Delete.                                                                                                        |
| <code>ui_type</code>                           | Type <code>text</code> into the focused field.                                                                                                              |
| <code>ui_inspect</code>                        | UI state: tool, workspace, panels, zoom, playback, selection, fps.                                                                                          |
| <code>ui_screenshot</code>                     | PNG of the window or one <code>panel</code> (Timeline, Program); <code>max_side</code> default 1600.                                                        |
| <code>ui_control</code>                        | Any control-channel <code>method</code> : <code>ui.set</code> , <code>ui.scroll</code> , <code>ui.playback</code> , <code>ui.panel.show</code> , ...        |

## 06 Example tool calls

Shapes follow the MCP input schemas; the ids are the ones returned in storyboard 1.

### Storyboard 1: import, build a sequence, lay the cut, add the lower third

```
{
  "name": "media_import",
  "arguments": {
    "paths": [
      "~/demo/stills/shot-a.png",
      "~/demo/stills/shot-b.png",
      "~/demo/stills/shot-c.png"
    ]
  }
} -> {"items": [1, 2, 3]}

{
  "name": "command_batch",
  "arguments": {
    "stop_on_error": true,
    "steps": [
      {
        "id": "file.newSequence",
        "params": {
          "name": "Rough cut",
          "width": 1920,
          "height": 1080,
          "fps": 23.976
        }
      },
      {
        "id": "timeline.place",
        "params": {
          "items": [1, 2, 3],
          "track": "V1",
          "seconds": 0
        }
      }
    ]
  }
} -> clips [11, 12, 13]

{
  "name": "command_run",
  "arguments": {
    "id": "graphics.newText",
    "params": {
      "text": "Dr. R. Young, UNLV",
      "position": [120, 960],
      "size": 72,
      "seconds": 5,
      "track": 2,
      "time": 0
    }
  }
}
```

### Storyboard 2: dissolve, Lumetri, preview frame

```
{
  "name": "command_batch",
  "arguments": {
    "steps": [
      {
        "id": "sequence.applyVideoTransition",
        "params": {
          "clip": 12,
          "effect": "cross_dissolve",
          "frames": 12,
          "edge": "in"
        }
      },
      {
        "id": "effects.apply",
        "params": {
          "clip": 13,
          "effect": "lumetri"
        }
      },
      {
        "id": "effects.setParam",
        "params": {
          "clip": 13,
          "effect": "lumetri",
          "param": "temperature",
          "value": 30
        }
      }
    ]
  }
}
```

```
  ]}}
  {"name": "render_frame", "arguments": {"seconds": 11, "max_side": 960}}
```

### Deliver: YouTube preset, burned-in captions, -14 LUFS

```
{
  "name": "command_run",
  "arguments": {
    "id": "file.exportMedia",
    "params": {
      "path": "~/Videos/out/final.mp4",
      "preset": "YouTube 1080p Full HD",
      "burnCaptions": true,
      "loudnessLufs": -14,
      "wait": true
    }
  }
}
```

## 07 Limits and tips

### ! Alpha: about 56% ready for real work

The project's own numbers: about 86% of Premiere's feature breadth, about 56% "ready for real work". No .prproj import (use FCP7 XML, AAF or OTIO), no VST3/AU/OpenFX plugins, H.264 is the only software delivery codec (HEVC via hardware encoders only, no AV1 export). Lumetri, keys and export rendering run on the CPU.

### ! Port 9876 has no authentication

FilmCraft (Live) accepts commands from *any* local process on 127.0.0.1:9876, including saves and exports to any path. Run it only while you are using it, quit it afterwards, and never forward the port.

### ✓ Work on copies, save often

Point Claude at a copy of the footage and a new .fcproj. The headless server starts with no project and keeps it in memory, so ask for `file.saveAs` before ending. Every `command_run` edit is undoable (`edit.undo`, `history.list`); `file.saveCopy` snapshots before risky batches.

### i Long jobs and missing media

Exports, proxies and scene detection run as background jobs: pass `wait true`, or poll `jobs.list` / cancel with `jobs.cancel`. Offline media renders as placeholder frames, so check `media.findMissing` first. Use absolute paths everywhere.

### i Optional downloads

Speech-to-text (Whisper tiny/base/small) and natural TTS voices (Kokoro-82M) are downloaded on demand, if installed: Whisper models via `transcript.downloadModel`, voices via `tts.downloadVoices`.

✓ **Same commands from the shell**

`filmcraft-cli describe <id>` prints one command's parameters; `filmcraft-cli exec <id> '<json>'`, `run script.jsonl` and `export out.mp4 --preset ...` are handy for cron jobs.