SmartMarkers

User Manual

PUBLIC BETA

Automatic song structure detection for lighting designers — REAPER markers, downbeat-accurate, with real confidence per section, for grandMA3 timecode workflows.

Public Beta. SmartMarkers is now publicly available — the closed testing phase is over, anyone can download the package and try it out.

No account, no licence key. You install it and start working. This test build runs until **01.01.2027**; after that the analysis stops working. The markers you have set by then stay in your REAPER projects — they live in your project files and are not affected.

Beta / early access. SmartMarkers is under active development. Song structure detection is mature and holds up in real work; some of the extra layers (see Chapter 6) are still work in progress. We say plainly what you can lean on and what is still growing up.

Feedback please by **email to beta@smartmarkers.tools** — the details are in Chapter 10.

1. What is SmartMarkers?

SmartMarkers analyses an audio file and puts **colour-coded, labelled song structure markers straight into REAPER** — intro, verse, chorus, bridge, mid-eight, outro and more. The boundaries between sections land **on the downbeat** (on the bar line, not somewhere near it), and every marker carries a **confidence value** that tells you how sure the tool is about that section.

Who is it for? Lighting designers who use REAPER as their **timecode master for grandMA3** (SMPTE/LTC or Art-Net). Listening through a track stays your creative work — that's where the ideas come from. SmartMarkers only takes the mechanical part off your hands: instead of hunting down downbeats frame by frame and nudging region boundaries around, you get a clean, bar-accurate structure and event grid in the arrangement after roughly half a minute per song. Your listening pass becomes pure idea collecting — with a lot more anchor points to hang light on.

You don't need to be **a programmer** to use SmartMarkers. The backend (the analysis software) is set up once and then runs quietly in the background — the actual work happens with a single keystroke in REAPER.

A REAPER arrangement after a SmartMarkers run: colour-coded structure Regions — Intro (blue), Verse (green), Chorus (orange), Bridge (purple), Mid Eight (orange/brown) — with confidence values above the waveform

---

2. What you get

After an analysis, REAPER shows you:

- **Colour-coded structure Regions** with a start and an end for every part of the song (intro, verse, bridge, chorus, mid-eight, outro …). Each section type has its own colour, so you take in the shape of the song at a glance.

- **Downbeat-accurate boundaries** — every Region starts exactly on a bar line, not a few milliseconds off. Exactly what you need for clean timecode cues.
- **Confidence per Region** (a value between 0.30 and 0.97). High values mean a solid detection. Lower values flag the sections you'll want to listen to yourself when you polish the show.
- **A curated set of show-event markers** — breakdown/break/stop with resume partners, drop, staged rises, strobe roll and drumroll, where detected.
- **The Beat Grid lane** — a neutral, white grid of every hit the analysis hears: your raw material. From there you promote whatever your show still needs with a single click (see chapter 5).

The event Lanes

Besides the song structure, SmartMarkers puts the detected events into **separate ruler Lanes** — on REAPER 7.62 and newer. The principle is simple: **one layer = one colour = one Lane of its own**. All Lanes carry the prefix "SM", so they never clash with other REAPER tools. The structure Regions get their own "SM Structure" Lane; related events deliberately share a Lane (Rise and Drop belong together, and so do breakdown/break/stop with their resume partners). The Lanes always appear in the same order, top to bottom: SM Structure, SM Bar, SM Roll, SM Strobe Roll, SM Rise/Drop, SM Breakdown/Break, SM Beat Grid.

The set is deliberately compact: the structure Regions, the show-event layers listed above, and the **Beat Grid** — a neutral white grid of every hit the analysis hears. The Beat Grid is your raw material: whatever your show still needs — an extra accent, a cue point of your own — you promote from there with a single click (chapter 5, the Smart Lane tool). The raw drum grids (Kick/Snare/HiHat/Beat) remain switched off by default.

Lanes are reused by name — running the analysis again updates the existing Lanes instead of creating duplicates. On older REAPER versions without Lane support everything still works, just without the separate Lanes.

All event layers are **point markers** ("something has to fire here") — only the song structure comes as **Regions** with a start and an end. Event markers the tool isn't sure enough about are never set in the first place.

Why this matters for lighting designers: every Lane is a clean cue family for programming in grandMA3 — one Lane maps to one sequence or light group. And because REAPER lets you show and hide Lanes individually, you can display only what you're working on right now — SM Structure + SM Breakdown/Break, for instance.

Lane	Colour	marks	Status
SM Structure	per section type	song structure as Regions (intro/verse/chorus…)	reliable
SM Bar	blue	bar grid (downbeats)	reliable
SM Roll	lilac	drumroll — the roll leading into an impact	curated; strongest on electronic music
SM Strobe Roll	purple	fast strobe-roll figures	curated; strongest on electronic music
SM Rise/Drop	cyan + pink	staged rises (start / drum / double), beat of silence, drop	curated; EDM/studio material

Lane	Colour	marks	Status
SM Breakdown/Break	dark/slate	breakdown, break, stop — each with a resume partner	curated; reliability varies by material
SM Beat Grid	white	neutral beat grid — everything heard without a confident name	the catch-all you promote from

Two quirks worth knowing. The **SM Breakdown/Break** Lane carries label pairs — the moment the rhythm cuts out, and "Resume" for the moment it comes back, both in the same Lane. The **SM Beat Grid** Lane deliberately makes no type claim: uniform white markers named "Hit", a complete grid for listening through and filling gaps — the right tool for that is the Smart Lane tool in chapter 5.

Switching Kick, Snare and HiHat on

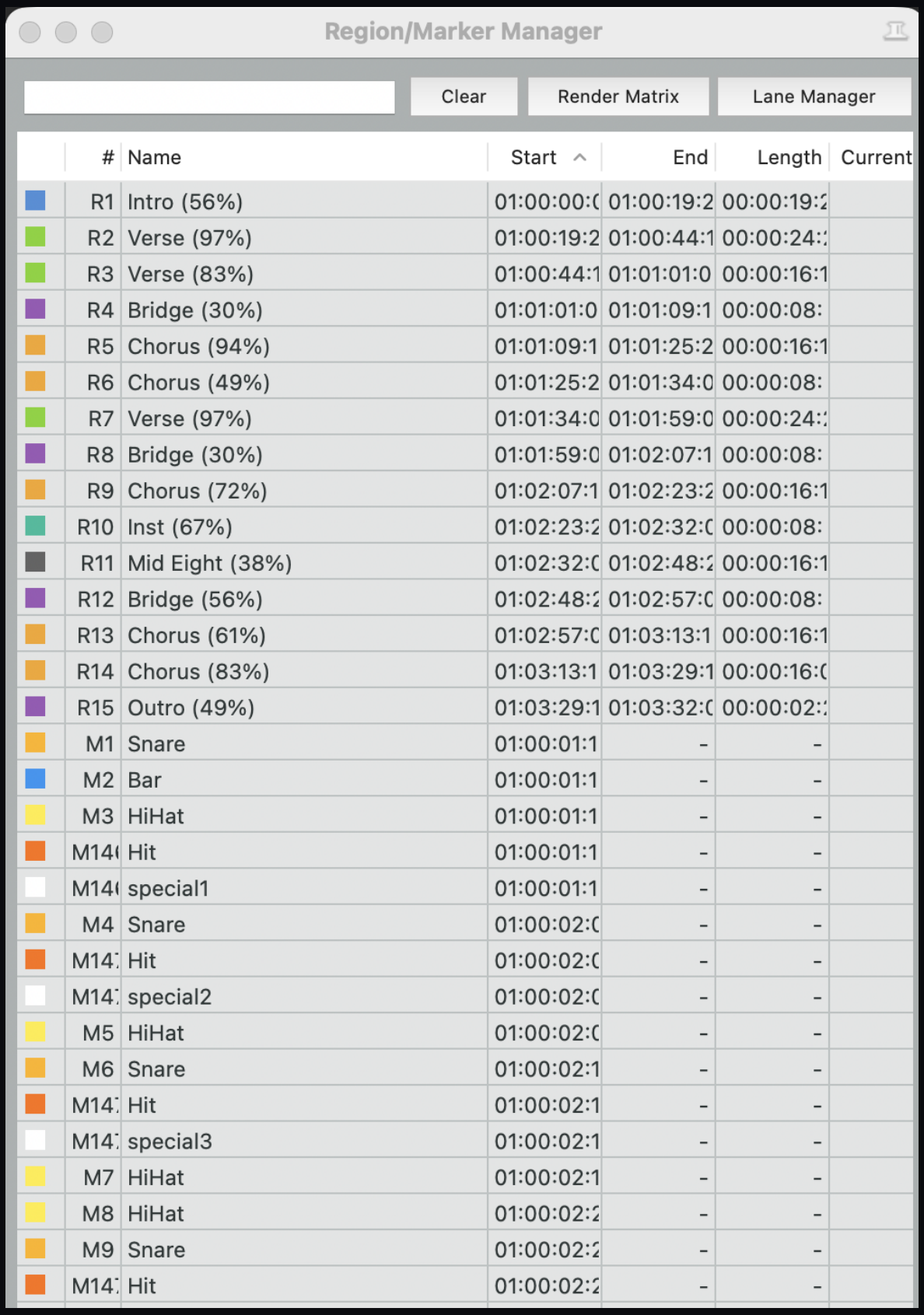
The three drum Lanes ship switched off (see above). If you need them — say for a continuous beat grid to program against — here's how:

1. Open `reaper/SmartMarkers.lua` in a text editor.
2. Find the `EVENT_LAYERS` section — quickest via your editor's text search for `EVENT_LAYERS`.
3. In the lines for `kick`, `snare` and `hihat`, change `enabled = false` to `enabled = true`. You can switch on just one of them if you prefer.
4. Save. The Lanes are included from your next analysis run onwards.

The same route switches on the even denser `beat` Lane (one marker per quarter note) — off for the same reason.

Two things worth knowing: first, you don't have to delete anything to get rid of them again — REAPER can simply hide Lanes (right-click the ruler → Ruler display → Ruler lane manager). Second, updating the script replaces the file, so your change is gone and has to be redone.

How much you see at once is up to you: with every Lane showing, the complete marker grid sits over the song — the Ruler Lane Manager lists each Lane with its marker count. Hide the grid Lanes and the arrangement clears up instantly, without a single marker being deleted.



#	Name	Start	End	Length	Current
R1	Intro (56%)	01:00:00:0	01:00:19:2	00:00:19:2	
R2	Verse (97%)	01:00:19:2	01:00:44:1	00:00:24:1	
R3	Verse (83%)	01:00:44:1	01:01:01:0	00:00:16:1	
R4	Bridge (30%)	01:01:01:0	01:01:09:1	00:00:08:	
R5	Chorus (94%)	01:01:09:1	01:01:25:2	00:00:16:1	
R6	Chorus (49%)	01:01:25:2	01:01:34:0	00:00:08:	
R7	Verse (97%)	01:01:34:0	01:01:59:0	00:00:24:1	
R8	Bridge (30%)	01:01:59:0	01:02:07:1	00:00:08:	
R9	Chorus (72%)	01:02:07:1	01:02:23:2	00:00:16:1	
R10	Inst (67%)	01:02:23:2	01:02:32:0	00:00:08:	
R11	Mid Eight (38%)	01:02:32:0	01:02:48:2	00:00:16:1	
R12	Bridge (56%)	01:02:48:2	01:02:57:0	00:00:08:	
R13	Chorus (61%)	01:02:57:0	01:03:13:1	00:00:16:1	
R14	Chorus (83%)	01:03:13:1	01:03:29:1	00:00:16:0	
R15	Outro (49%)	01:03:29:1	01:03:32:0	00:00:02:1	
M1	Snare	01:00:01:1	-	-	
M2	Bar	01:00:01:1	-	-	
M3	HiHat	01:00:01:1	-	-	
M14	Hit	01:00:01:1	-	-	
M14	special1	01:00:01:1	-	-	
M4	Snare	01:00:02:0	-	-	
M14	Hit	01:00:02:0	-	-	
M14	special2	01:00:02:0	-	-	
M5	HiHat	01:00:02:0	-	-	
M6	Snare	01:00:02:1	-	-	
M14	Hit	01:00:02:1	-	-	
M14	special3	01:00:02:1	-	-	
M7	HiHat	01:00:02:1	-	-	
M8	HiHat	01:00:02:2	-	-	
M9	Snare	01:00:02:2	-	-	
M14	Hit	01:00:02:2	-	-	

The Region/Marker Manager listing every Region that was set (R1–R15, Intro … Outro with start/end/length) and the markers with their timecodes — handy for jumping around and checking

3. Requirements

Before you start you need:

- **REAPER** — any reasonably current version.
- **A Mac with Apple Silicon** (M1 or newer) — mandatory; **Intel Macs are not supported**. Plus **macOS 14 or newer**. The installer checks both right at the start and stops on unsuitable Macs with a clear message instead of failing halfway through. **No** dedicated graphics card and **no** CUDA required — the analysis runs on Apple Silicon via MPS acceleration.
- **Nothing else.** The installer uses your existing Python 3.11 if your Mac already has one. If it doesn't, the installer offers to install it via Homebrew (defaults to no — Homebrew stays on your system for good). If you decline, or Homebrew isn't there, it asks right afterwards whether it should download a self-contained, standalone Python (~26 MB) straight into the SmartMarkers folder — **that question defaults to yes**, so Enter is enough. No manual install, no admin password, no Terminal needed.
- **Free disk space**: up to **8 GB** for the base installation (Python environment, analysis software, model data, cache) — that is exactly the threshold the installer checks before it starts. The optional second detector (see chapter 4) needs another **1.1 GB** or so.
- **Internet on the first run**: the detection engine itself (SongFormer) downloads around **3 GB** of model data once, during the very first analysis, and stores it locally. After that everything runs offline.

Short version: you already know REAPER. The rest is a double-click, and

from then on SmartMarkers is a keystroke.

---

4. Installing, step by step

Installing takes two double-click steps: **(A)** run the installer once, **(B)** register the SmartMarkers script in REAPER. That's it — you won't have to think about the backend after that.

A) Run the installer (once)

Double-click `Install-SmartMarkers.command` in the SmartMarkers folder.

The very first time you open it: macOS blocks unsigned scripts. >

macOS 14 or older: right-click the file and choose **"Open"** — then

confirm "Open" in the dialog. After that a double-click is enough. >

macOS 15 (Sequoia) or newer: Apple removed that right-click route.

Double-click (the blocked-file message appears) → close the dialog →

System Settings → **Privacy & Security** → the blocked file is listed at the

very bottom → click **"Open Anyway"** → confirm in the dialog. >

Important: this approval is needed for **each of the three `.command`

files individually** — so up to three times, the first time you open `Install-SmartMarkers.command`, `Start-SmartMarkers.command` and `Stop-SmartMarkers.command` respectively. After that a double-click is enough everywhere.

The installer sets everything up on its own: it uses your existing Python 3.11 if one is already installed. If not, it offers to install it via Homebrew (defaults to no). If you decline, or Homebrew isn't there, it asks right afterwards whether to download a standalone Python (~26 MB) straight into the SmartMarkers folder — **that question defaults to yes**, one press of Enter is enough. No manual install, no admin password. It then creates a sealed-off environment inside the SmartMarkers folder and installs the analysis software. That takes a few minutes; at the end it reports **"Installation complete"**. The rest of your system is left alone.

Along the way the installer asks where the roughly 3 GB of analysis models should go — with three options:

1. **Into the SmartMarkers folder** (recommended, just press Enter) — everything in one place, and deleting SmartMarkers takes the models with it.
2. **Into the standard system folder** (`~/.cache/huggingface`) — handy if other AI tools on your Mac use the same models.
3. **A folder of your own**, on an external drive for example.

If you run the installer again later, it only asks "keep this?" instead of offering the full choice again. If it can no longer find the folder you saved — because you renamed or moved the SmartMarkers folder — it says so and falls back to the SmartMarkers folder instead of re-creating an empty one in the old place.

Extra step: more accurate detection (optional). At the end the installer offers to set up a **second, independent detector**. It cross-checks the song sections: where the two agree, the boundaries sit far more reliably (measured across 32 songs: 40 % confirmed boundaries instead of 26 %). For that the installer creates a second Python environment — a one-time download of about **250 MB**, taking a good **1 GB on disk** afterwards; your first analysis run then fetches another ~100 MB of model data. The question defaults to **yes** (`[Y/n]`).

Optional means optional: if you decline — or if the step fails, say because your connection drops — you have a fully working installation, just with one detector instead of two. Your main installation is never affected. **You can add it later at any time: simply run `Install-SmartMarkers.command` again** — it recognises everything that is already installed and skips it.

Language. SmartMarkers talks to you in **English** — the installer sets that up automatically, regardless of your system language. There is nothing to choose and nothing to configure.

The accompanying documents live in the `docs/` folder. **Recommended starting point: open `docs/SmartMarkers_Overview.html` — everything is linked from there.** The full set:

- `SmartMarkers_User_Manual.pdf` — this manual as a PDF
- `SmartMarkers_Manual.html` — the same manual as HTML
- `SmartMarkers_Beta_Log_EN.html` — the beta log (reported bugs and what came of them)
- `SmartMarkers_Rules.html` — the rules the markers follow, plus a questionnaire

- `SmartMarkers_Feedback_EN.html` — the feedback form (see chapter 10)
- `SmartMarkers_Complete.html` — all documents as tabs in one single file

The SmartMarkers folder itself also contains `README.txt` with the quick start.

About that first run: during the first real analysis SmartMarkers downloads around 3 GB of model data (the SongFormer weights) once and puts it wherever you chose above. Plan for up to 8 GB of free disk space for the base installation including the models (plus around 1.1 GB if you also set up the second detector).

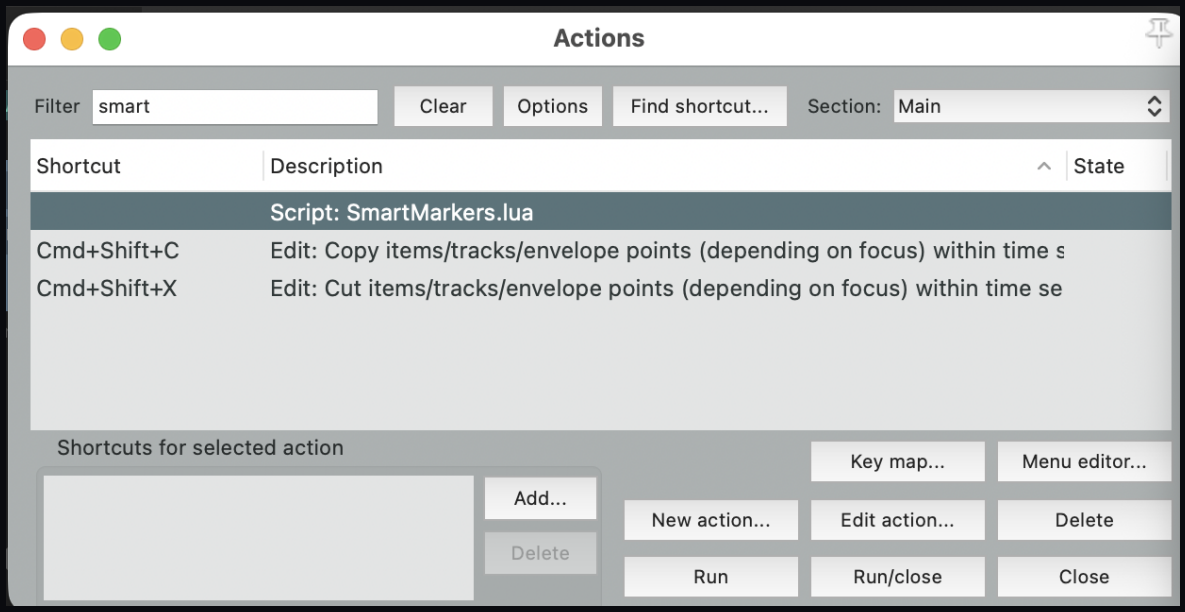
This happens only once — after that the analysis starts without any further download. If the download is interrupted (network drops, lid closed, you cancel), it picks up where it left off next time instead of starting over.

B) Register the SmartMarkers script in REAPER

You do this **once** in REAPER:

1. **Actions** → **Show action list** (shortcut `Shift + /` on US keyboards; on German keyboards that's `Shift + 7`. If you can't find the shortcut, just use the menu: **Actions** → **Show action list**.)
2. Click **"New action…"** → **"Load ReaScript…"**
3. Pick the file `reaper/SmartMarkers.lua` from the project folder.
4. Optional but recommended: assign the action a **keyboard shortcut** — `Shift + M`, for example.

That makes SmartMarkers available as a REAPER action. **No** extra Python configuration inside REAPER is needed — the script talks to the backend itself.



The REAPER Action List, narrowed down with the filter "smart" and the highlighted row "Script: SmartMarkers.lua" — this is where you run the action or assign it a keyboard shortcut

C) The backend? Not something you need to remember.

The analysis backend **starts by itself** as soon as you run SmartMarkers in REAPER and it isn't running yet. The first start of the day can take up to a minute (the console shows the progress) — after that it's there instantly.

If you prefer doing it by hand: double-clicking `Start-SmartMarkers.command` starts the backend in a window of its own. In case something goes wrong, the auto-started backend writes its log to `~/Library/Logs/SmartMarkers/backend.log` (older builds: `/tmp/smartmarkers_backend.log`).

D) Shutting the backend down

Once you've used it, the analysis backend keeps running in the background (it holds the model in memory so the next run starts fast). That's deliberate: it runs **independently of REAPER** on purpose and therefore stays alive when you quit REAPER — so the next run is fast again, and quitting REAPER can't kill an analysis in progress.

It does occupy several gigabytes of RAM while it sits there, though. That's why, since v0.1.0-beta.6, it **shuts itself down after 30 minutes without an analysis** — and starts up again automatically on your next `Shift + M`.

If you want it gone right now (big project, rendering, memory getting tight): double-click `Stop-SmartMarkers.command`. The script first checks that the thing answering on the port really is the SmartMarkers backend — it won't touch another program — and then shuts it down cleanly. If an analysis is running, it is allowed to finish; up to a minute of waiting is normal here. If you started the backend by hand via `Start-SmartMarkers.command`, `Ctrl+C` or simply closing that window does the job too.

---

5. First run / workflow

Once the script is registered, the routine is quick:

1. **Load the audio file into REAPER** — as an Item on a track.
2. **Click the Item** so that it is selected (you'll see the brighter border). This matters: SmartMarkers analyses exactly the Item that is selected.
3. **Trigger the SmartMarkers action** — with your shortcut (`Shift + M`) or from the Action List.

From here it runs by itself: the script checks whether the backend is up (and starts it if needed), sends the audio file off for analysis, removes **all markers and regions in the project** and puts the new structure into the arrangement. The **ReaScript console** shows you every step (detected tempo, number of markers, duration, and whether the analysis ran "… via SongFormer" or "… via librosa fallback"). Reckon with roughly **20–35 seconds per song**; while the analysis runs REAPER won't respond — that's normal, there is no progress bar.

Important: before setting anything, SmartMarkers removes **all** markers and regions in the project — not just its own. Since beta.3 that happens together with setting the new markers in **one** undo step: `Cmd+Z` reverts

both. Still, **don't** run SmartMarkers in projects that contain your own markers that matter without saving a copy first — better safe than sorry. If you only want the visible SmartMarkers Lanes gone without re-analysing, `reaper/SmartMarkers_DeleteVisible.lua` is included for that (it deletes the currently visible marker Lanes — explained later in this chapter).

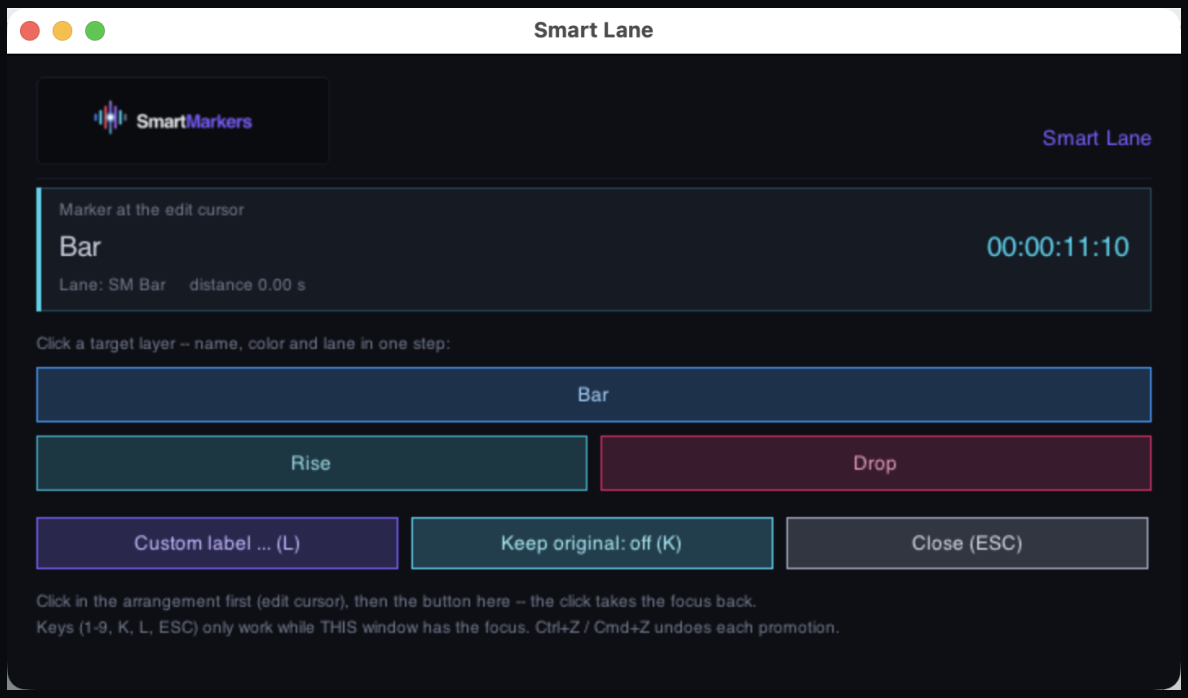
After that your song structure sits in the arrangement as a colour-coded grid of Regions.

Working through the Beat Grid — the Smart Lane tool

The white **Beat Grid** lane is your safety net: as complete a grid as possible of every hit the analysis heard. If your show is missing a marker — a drop, a break, a cue point of your own — you don't rebuild it by hand, you promote the matching Beat Grid marker with **Smart Lane**:

1. **Register `reaper/SmartMarkers_SmartLane.lua` as an action** (once, the same way as the main script) and run it — the **Smart Lane** window in the SmartMarkers design opens and stays open.
2. **Click the marker in the arrangement** (that sets the edit cursor). The window immediately shows the nearest marker's name, timecode and lane.
3. **Click the target layer in the window** — "Drop" or "Strobe Roll", say. Name, colour and lane change in one step (and in one undo step: `Cmd+Z` reverts the whole thing).

Two switches live in the window as well: **[Custom label …]** sets a free marker name (colour and lane still come from the target), **[Keep original]** leaves the Beat Grid marker in place instead of replacing it.



The Smart Lane window: at the top the marker at the edit cursor with name, lane and timecode; below it the target layers (Bar, Rise, Drop, Breakdown, Break) and the switches Custom label, Keep original and Close

One habit worth building: click in the arrangement first, then the button

in the window. A click into the arrangement hands REAPER the keyboard focus —

keyboard shortcuts only work while the tool window has focus; clicking a

button takes it back automatically.

Clearing out Lanes — the delete tool

`reaper/SmartMarkers_DeleteVisible.lua` (register it once, the same way) deletes every marker and region that is **currently visible** in the ruler — whatever sits in a hidden Lane stays untouched. That turns REAPER's Lane visibility into a precise delete filter:

1. In the **Ruler lane manager** (right-click the ruler → Ruler display → Ruler lane manager), **hide the Lanes you want to KEEP** (switch their eye icon off).
2. **Run the script** — everything still visible is deleted, as **one** undo step (`Cmd+Z` brings it all back).
3. **Unhide the Lanes** again. What you hid in step 1 is all that's left.

Typical use: the show is programmed, and you only want to keep SM Structure and SM Breakdown/Break — hide those two, run the script, unhide, done.

Exporting markers as CSV

SmartMarkers puts its markers straight into REAPER — a CSV export is REAPER's own built-in function, not a SmartMarkers feature: open **View** → **Region/Marker Manager**, **right-click** in the list → **Export…**, and pick **CSV** as the file type. From there you can prepare the file for your lighting console with whatever tool you like.

Pairing with grandMA3

REAPER stays your **timecode master**: it sends timecode straight to grandMA3 (SMPTE/LTC or Art-Net). The Regions SmartMarkers sets are your cue reference grid — bar-accurate points in time for intro, chorus, bridge and so on, to hang your looks and executor cues on. SmartMarkers changes nothing about your existing timecode setup; it just hands you the structure you build the show around.

---

6. What's solid, what's still beta

SmartMarkers is honestly scoped, so you know what you can rely on and what is still maturing. There's a Lane-by-Lane breakdown in the event Lane table in chapter 2; here's the short version:

Reliable and ready to use

- **Song structure detection.** The core of it. Intro, verse, chorus, bridge, mid-eight, outro and other sections are detected reliably — including short transitions such as a bridge right before a chorus, or a mid-eight inside an instrumental block, which a lot of tools miss.
- **Downbeat-accurate snapping.** Every section boundary sits exactly on the bar line — precisely what matters for clean timecode work.
- **Real confidence values per marker.** The confidence you see isn't decoration; it comes straight out of the detection model. Where SmartMarkers is unsure it tells you — boundaries reconstructed

heuristically deliberately carry lower values.

- **The Beat Grid.** The neutral beat grid claims nothing it doesn't know — which is exactly why you can rely on it: as a complete reference grid for listening through and promoting from. The fine-grained drum grid (Kick/Snare/HiHat) is still there but ships switched off — see "The event Lanes" for how to switch it on.

Under active development (beta / work in progress)

These layers are built in already, but they aren't meant as **finished features** yet — treat their results as a hint, not as something to rely on:

- **The curated event layers (breakdown/break/stop/resume, drop, rise, rolls).** They have held up in our reference comparisons — strongest on electronic music; on live band mixes some layers are still noticeably less reliable and are being actively developed. Your feedback on exactly these layers (chapter 10) feeds straight into the next round.
- **Vocal detection.** Spotting vocal entries is still limited in dense live mixes and isn't production-ready right now.

Rule of thumb: rely fully on the song structure markers and the confidence values for your structural backbone. Treat the experimental rhythm and vocal layers as useful extra hints that you double-check when it matters.

A note on file formats

For the best results use **WAV** rather than MP3. MP3 files can introduce a small time offset (roughly 20–40 ms) that can shift the bar-accurate placement slightly. Supported formats: **WAV (recommended), AIFF/AIF, FLAC, M4A, OGG, MP3** (MP3 with the caveat above) as well as **video Items** (MP4/MOV/M4V and others — the audio track is extracted automatically).

---

7. Troubleshooting

The analysis produces no markers / nothing happens at all:

- Open REAPER's console via **View → Show console window**. SmartMarkers logs every step there — error messages show up there first.
- Is an audio Item really **selected**? SmartMarkers needs a selected Item (brighter border).

Error message "backend could not be started":

- Normally the backend starts automatically. If that fails, take a look at the log file `~/Library/Logs/SmartMarkers/backend.log` (in older builds: `/tmp/smartmarkers_backend.log`) — the actual error message is in there.
- Most common cause: an incomplete installation. Just run `Install-SmartMarkers.command` again (the installer works out what's missing and repairs it).
- Alternatively you can start the backend visibly: double-click `Start-SmartMarkers.command` — then you see the errors right in the window.

Port 8342 is occupied / a message about another program:

- SmartMarkers talks to its backend over local port **8342**. On start it checks what is answering on that port — three cases are possible: the port is **free** (the backend is started automatically), its **own backend is already running** (it is simply reused), or **another program** occupies the port — in which case the REAPER console shows a clear message naming the blocking program, instead of mistaking its answers for SmartMarkers' own.
- The fix in the third case: quit the named program (or configure it to use a different port) and trigger SmartMarkers again.
- `Stop-SmartMarkers.command` deliberately does **not** help here: it only shuts down SmartMarkers' own backend and never touches another program on the port.

The analysis hangs or takes forever:

- Is this the **very first run**? Then SmartMarkers is downloading ~3 GB of model data once (plan for up to 8 GB of free disk space for the base installation including the models) — depending on your connection that can take a while. REAPER doesn't respond during an analysis (no progress bar), which is normal. Every later run is fast (~20–35 s per song).
- Check REAPER's console: after each analysis it reports "… via SongFormer" or "… via librosa fallback". If it says librosa fallback, only the simpler stand-in mode is working right now (it functions, but at lower quality) — usually that means the model data isn't fully downloaded yet or the installation didn't complete; running `Install-SmartMarkers.command` once more fixes it. If in doubt, the log file has the details (see above).
- **Where exactly the model data lives** is in the log file too: the backend writes the effective model folder once at startup (line "[songformer-warmup] Modell-Ordner: …").

Markers sit slightly off / quality feels weaker:

- Use **WAV instead of MP3** (MP3 time offset of 20–40 ms).
- Use the console to make sure SongFormer is active (see above).

---

8. Removing SmartMarkers

Honestly: deleting the SmartMarkers folder isn't **quite** enough. Depending on what you picked during installation, up to **8 GB** can sit outside that folder — model data, an analysis cache, a log file. Here is point by point what lives where.

The points are independent of each other, so you can clean up just some of them. **None of this deletes your REAPER projects or the markers you've set** — those live in your project files and stay put.

First: double-click `Stop-SmartMarkers.command` so nothing is running

in the background while you clear up.

a) The SmartMarkers folder itself (~2 GB, around 3.1 GB with the optional second detector)

Drag the folder to the Trash, done. It holds the sealed-off Python environment with the analysis software (plus, if you set it up, the second environment of the more accurate detection, ~1.1 GB) — and, if you picked the recommended option 1 during installation, the model data and the cache as well. In that case

you're already finished here and can skip ahead to point (d).

b) The analysis models (~3 GB, plus the ~100 MB of additional models the second detector fetches on its first run)

Where they live depends on what you chose during installation:

- **Option 1 (SmartMarkers folder)** — already handled by point (a).
- **Option 3 (a folder of your own, e.g. an external drive)** — delete that folder. Which one it was is written in the file `backend/MODELS_DIR` inside the SmartMarkers folder (if you haven't deleted it yet).
- **Option 2 (standard system folder)** — the models are in `~/.cache/huggingface` and `~/.cache/torch`.

Careful with option 2: those two folders are **shared** — other AI programs on your Mac (transcription, stem separation, image tools) put their models there too. So don't throw them out wholesale if you use software like that.

Have a look at what's actually in there first:

```
ls ~/.cache/huggingface/hub
ls ~/.cache/torch/hub/checkpoints
```

The ones belonging to SmartMarkers are `models--ASLP-lab--SongFormer` (the big structure model), `models--taejunkim--allinone` and the files with `beat_this` in the name. Everything else belongs to other programs.

To remove only the SmartMarkers models:

```
rm -rf ~/.cache/huggingface/hub/models--ASLP-lab--SongFormer
rm -rf ~/.cache/huggingface/hub/models--taejunkim--allinone
rm -f  ~/.cache/torch/hub/checkpoints/beat_this*
```

If you use **no** other AI software at all, you can simply delete both folders instead (`rm -rf ~/.cache/huggingface ~/.cache/torch`).

c) The analysis cache (up to ~4 GB)

SmartMarkers stores intermediate results so the same song finishes faster the second time. With options 1 and 3 this cache sits in the model folder you chose (so points (a) and (b) cover it); with option 2 it's here:

```
rm -rf ~/.cache/smartmarkers
```

That folder belongs to SmartMarkers alone — you can delete it safely, also in between whenever you need the space.

d) The log file (a few MB)

In one of two places depending on the version — neither command does any harm if that place doesn't exist:

```
rm -rf ~/Library/Logs/SmartMarkers
rm -f  /tmp/smartmarkers_backend.log
```

e) Unregister the action in REAPER (please don't forget this one)

Otherwise a dead entry stays in the action list — along with your `Shift + M`, which then grabs at nothing:

Actions → **Show action list** → filter for `SmartMarkers` at the top → select the line `Script: SmartMarkers.lua` → `Remove` (confirm the prompt). The shortcut goes with it.

f) Optional: Python

- If the installer downloaded a **standalone Python** into the SmartMarkers folder (the usual case if you had no Python 3.11 of your own), point (a) has already taken care of it.
- If you went with **Homebrew** during installation instead, that Python 3.11 stays on your Mac permanently.

In the second case, this is how you remove it — but only if you're sure **no other program** uses it:

```
brew uninstall python@3.11
```

When in doubt, just leave it: it takes up little space and gets in nobody's way.

---

9. Licence & credits

SmartMarkers is **proprietary software** — all rights reserved. For the public beta the enclosed `BETA-LICENSE.txt` applies: a personal, non-transferable right of use for the duration of the test phase, with no right to pass it on. Please don't share either the software or this manual.

The detection engine at the heart of it is **SongFormer** by the ASLP Lab. It is **licensed separately under CC-BY-4.0**. That means: any use or distribution that includes the model or its output requires **attribution** to SongFormer / ASLP Lab.

Also part of the detection: **beat_this** (CPJKU, 2024) for frame-accurate beat and downbeat detection, and **librosa** as the fallback analysis path.

---

10. Feedback

In the public beta, feedback runs through a single channel: **email to beta@smartmarkers.tools**. There are two main paths — take the one that fits your finding:

1. **Feedback form (for bugs, usability, ideas).** The package includes the form `docs/SmartMarkers_Feedback_EN.html`. It runs in the browser, **sends nothing by itself** and only produces a block of text — copy it and email it to **beta@smartmarkers.tools**. One form per finding.
2. **Red markers (for everything marker-related).** After the analysis, drop a red default marker at the spot in question, write your note into the marker text, and send the REAPER project (.rpp) plus the audio file as a ZIP to the same address. **Moving or deleting** markers counts as feedback too — comparing against the original shows us exactly what you corrected.

Short things are welcome as a plain email — the version number (shown in the REAPER console) helps us match your report to a build.

Most valuable of all: a song you have already marked up by hand. You'll see immediately where the detection agrees with you and where it doesn't — and that comparison is what helps me most.

Training material

If you want to help improve the detection: send me **3 to 10 of your REAPER projects** with the markers you set and the matching audio file to **beta@smartmarkers.tools** (for large files a link to your cloud storage is welcome). I let the model analyse the same files and compare its markers with yours — that's how it learns the decisions you actually make in practice.

EDM in particular is still weak: risers, build-ups, breaks and drops.

The material stays with me, is used for training only, and is not passed on.

SmartMarkers is beta software under active development. Feedback from real shows is what helps us get the experimental layers to maturity faster.