# AudioCortex 2

Help · version 2.5.49

This book is the app's own help, the help of the build that made it: every room, then the controls and the things Oscilla (the app's agent) can be asked to do in that room.

Iced Audio

# Contents


## Agent Activity

Re-create the steps · The agents' history

## Arranger

Add chord presets · Answer a note · Arrange song · Change one section · Compose line · Fill the chord pads · Harmonize a melody three ways · Hide the Song Map · Hide the panel · Job status · Lisp arrangement forms · Oscilla's inbox · Project add track · Project bounce · Project list · Project open · Project save · Projects and saved versions · Read the Song Map and what you point at · Set the Song Map · Show or hide the Song Map · Show the Chords tab · Song bounce · Song bounce arc · Song start · Song stop · Suggest a progression · Synth sequence · The Chords tab · The harmony rules · The note two parts disagree on · The pads' sound (tool) · What Oscilla is working on · What chord comes next

## BPM Tap

## Delay Finder

## DJ

DJ break · DJ crate · DJ crate read · DJ crossfade · DJ drop · DJ eq · DJ load · DJ play · DJ status · DJ suggest · DJ sync · DJ tags: tempo, key, energy · Export a playlist to the decks · Record the decks

## Instruments

Drum bounce · Drum hit · Drum kit · Drum pattern · Drum sample · Drum stop · Drum tune · FM bank · FM patch · FM preset · Grow patch · Pd control · Pd load · Pd record · Pd stop · Poly voice · Synth note · Synth param · Synth preset · Synth render · Wave program · Wavetable

## Library

A loop at the session's tempo · Activate AudioCortex · Add to a scan set · Add to the Crate · An index beside the library · Analyze · Ascension · Bones · Browse folder · Chromagram: notes over time · Classify · Coming from AudioFinder · Control AudioCortex from your phone · Convert files · Create collection · Descriptors · Detect BPM · Detect key · Embedded metadata (AudioFinder's) · Envelope · File metadata · Find by category · Find duplicates · Find files · Find similar · Find sounds on the web · Finder Selection · Freeze · Get annotation · Get selection · Glossary: list, collection, scan set, crate, analyze, classify, deep scan, MIDI index · How fresh an index is · Import a sound from the web · Import raw PCM · Library similar · List files · List locations · List scan sets · Load scan set · Loudness · Loudness over time · Lush stretch (a dense, wide cloud from a recording or from the synth) · MIDI In · Melody from a sung recording · Model categories · Morph · Mosaic · Move scratch out of Recordings · Onsets on the bar grid · Open window · Pitch · Play · Play AudioCortex from a MIDI controller · Process · Recordings, songs and what a song keeps · Remove from the Crate · Resonate · Reveal in the Finder (tool) · Reverb · Rooms and shortcuts · Scratchpad · Search · Search by text · Search every analyzed sound · Search the index · Sediment · Select file · Set comment · Set rating · Settings · Slice at beats · Slice file · Spectral cross · Spectrogram on an absolute scale · Spectrum · Spectrum live · Speed · Status · Stereo field over time · Stop · Tags on sounds · Texture fingerprint (what makes a recording sound the way it does) · The Metadata window · The audio editor · The sound index · Tonal balance against references · Veil · Vocal remove · Waveform image · Wavesets · What Recordings holds · What is analyzed, folder by folder · Widen · Workspace · Workspaces: which rooms the rail shows · Write embedded metadata

## MIDI Library

MIDI perform · MIDI read · MIDI scan · MIDI search

## Musical Judge

Aesthetic · Judge · Judge musical · Sounds like

## Nodes

A render, rebuilt from its recipe · AudioUnit validation · Every kind of Nodes card, described · Generate a part · Hear a proposal · Mix Narrowcaster: the small mono speaker, the translation check · Multisamples and SoundFonts · Nodes get · Nodes plug-in · Nodes preset · Nodes presets · Nodes render · Nodes set · One piece back · Proposals · Set a proposal aside · Take a proposal · The pieces of a song · The proposals · Tidy the Nodes canvas · What each stage of a chain changes

## Oscilla

A hardware instrument behind a lane · Agent note · Agent notes · Clear the knobs · Help · Knob Panel · MIDI clock to a device · MIDI learn · Map a control · One filed report (tool) · Oscilla Local (the downloaded model) ·

Read the knobs · Report a Bug · Report a bug (tool) · See the controllers and mappings · Send to your hardware · Session note · Session status · Stop all · Watch what a device sends · Welcome: Oscilla and requirements · When the app froze · Your filed reports (tool)

## Sounds

Aurora presets · Mystical presets · Nova presets · Sounds browse

## Spiral

## Synthesizer Sampler

Connect the sampler · One note, read back · Record the library · The libraries made · What the run will record · What the sampler can connect to · Where the run is · Who is the synth

## The Explorer

Explorer configure · Explorer show

## Tone Finder

## Tracks

A lane’s fader and pan (tool) · A lane’s send (tool) · A mix onto the lanes · A mix, kept by name · A take back on its lane · A track's structure, from the file · Act on whole lanes · Analyze mix · Automate a project lane · Balance start · Bounce the Tracks room · Change a lane · Channels and plugins · Checkpoints: keep a moment of the song · Clickiness · Clips on the lanes · Collect all files into the song · Compare two files · Concat files · Do the stems line up? · Draw an automation line · Drop files onto a track · Ducking between lanes · Edit a clip · Ember EQ: the character EQ for every lane, the master and the canvas · Ensemble · Export to Ableton Live · Glitch finder · Group lanes (tool) · Hide the Sounds tab · Import a Live set (tool) · Import an 0RKESTRACKER song (tool) · Lane height · Level waves · Listen to anything playing · Listening, now · Listing the song's checkpoints · Locking a decision · Loop a region · Master and release · Missing files in Tracks · Mix files · Mix map of a project · Name a lane or the set · Notes on a lane · Play layers · Play the Tracks room · Project remove track · Record in Tracks · Reinforce a stem pack · Render a song from a score · Restoring a checkpoint · Retitle a project · Return lanes (tool) · Section balance · Show a lane's line · Show the Sounds tab · Start a new song · Stems from the Tracks room · Stems reinforce · Stereo width, band by band · The Sounds tab · The Sounds tab, read · The Tracks tempo · The automation lines · The beat grid · The ear meter: what is heard of each part · The groove, folded onto a bar · The hand mix moves · The lanes of a project · The live mix, in one call · The melody, note by note · The mixes kept · The sections, from the bands · The takes, kept · Timeline analysis · Timing against the grid · Tonal lines · Tuning from the partials · Two mixes, compared · Two moments, compared · Undo a clip edit · Warp to a grid · What came from the phone · What was heard · Where a project's lanes are kept · render_cancel — stop a long render · render_jobs — the long renders running now

# Agent Activity

## Agent Activity

Developer builds only: not in the Help menu of the app you downloaded. The developer activity view shows agent tool calls as they happen — timing and success, who is driving. The same calls are written down for good in ~/Documents/AudioCortex/History (one JSON line each, appended, never rewritten) and in a project's own History folder; the history tool reads them back and history_replay runs the steps again.

Ask Oscilla: history, history_replay, session_status

## Re-create the steps

The history can be run again. history_replay takes a project's record (or a day's) and calls the same tools with the same arguments, in order — only the calls that worked, never the history tools themselves, and never a replay of a replay. Bound it by line (from, to), keep only some tools (tools) or leave some out (skip); dry lists the steps first. It stops at the first step that fails unless told to continue. Every replayed call is recorded again, marked as a replay, so the record stays honest about who did what.

Ask Oscilla: history_replay, history, project_open, project_tracks

## The agents' history

Everything an agent does is written down as it happens: one line per tool call — the time, the client that asked (Claude Code, Oscilla, a script), the tool, its arguments, whether it worked, how long it took, and the reply (an image kept as its size, text to 8 KB). The lines are JSON, one per line, appended and never rewritten, in ~/Documents/AudioCortex/History/<day>.jsonl — and a call about a project (one it names, or one made while a project is open) is also kept in the project's own History/actions.jsonl with a note of whether the step changed the project. The history tool reads them back: a project's, a day's, one tool's, one client's, since a time. Every file of the song is also written through one door, and each write leaves a receipt in the song's History/receipts.jsonl: which part changed, the file, and who caused it — an agent's tool by name, your own hand as you; each change is also written to History/journal.jsonl the moment it is made, before the file, so who changed what is known change by change. A change made from your own mouse or keys is credited to you; one nobody claimed is marked unknown and protected like yours; a song file changed outside the app is recorded as external when the song is next opened. Nothing in it is a sound file or a license code. The same record is a window: File ▸ History… lists the song's changes newest first — yours by hand as what moved from → to, the agents' calls, the phone's presses and the locks — filtered by author, with Replay to here for an agent's step and a door to Moments.

Ask Oscilla: history, history_replay, project_add_track, project_open, project_save

# Arranger

Plan with arrange_song; compose_line supplies rows and song_bounce_arc supplies changes.

## Add chord presets

Preset rows under the pads load a progression into the steps with one tap. The built-in ones are the shapes the arrangement recipes use, spelled in the current key; Oscilla can add its own, named by how they sound.

Ask Oscilla: chords_set_presets, chords_get

## Answer a note

A short reply to one of your notes, shown under the note in the Song Map and the inbox; done (the default) closes it. The words are for her: what was done, in a line.

Ask Oscilla: inbox_reply, inbox, song_map_get

## Arrange song

Ask Oscilla for a song in a style and it plays it live in the Arranger, looping until you press Stop, so you can change a lane's sound or mute it while it plays. Nothing is recorded until you ask: say record or bounce (or press Record) and one pass is captured as stems, a mix and a project. A record longer than about forty seconds comes back as a job that finishes on its own; job_status follows it. Asking again while it plays restarts with the new recipe. `background` true returns a job id at once; poll job_status.

Ask Oscilla: arrange_song, project_list, job_status

## Arranger

The Arranger shows instrument rows, drums and section changes. Ask Oscilla for a style, key and tempo, inspect the plan, then play or record it. A sequence is a row of sixteenth notes; four beats make a bar. Recording takes real time. A titled arrangement saves a project you can reopen. Play plays the recipe live and loops it until Stop — nothing is written — and pressing it again restarts with the recipe as it is now. Record captures one pass as a project. Stop stops at once; a recording keeps what was played. Space plays and stops, as Play and Stop do, whenever no text field is being typed in. When a recorded take ends, its voices and drums fall silent on their own. Each lane's sound picker opens on the kind of sound the lane has now (Analog, FM or Tables). A sound kept for a voice the current style does not have is dropped from the recipe rather than failing the take. An empty Arranger says so: ask Oscilla for a song (three examples are there to click), or press Play to hear the recipe in the strip. In the recipe strip an empty BPM field (auto) means the style's own tempo and an empty seed means a new one each take. On the timeline, changes that land on the same bar share one marker and one label. BPM and MODE read auto when the style chooses them. While a song plays, a lane with no notes yet says when it comes in (or, should a part never come in, says so). Changes that fall after the last bar are left out of the take. A take shorter than a style's arc is the arc in miniature: its sections are scaled to the take's length on a four-bar grid, so every part comes in before the take ends; sections that land on the same bar are applied in order — a form that writes past the take's end is scaled the same way. A new song clears the stories a hand-made line left on its lanes. Each lane has M to mute and S to solo (hear it alone; S again hears everything), a level knob beside its meter, and – 8va + to shift it by octaves; OCTAVE in the transport row shifts every lane. Changing the style, key, mode, tempo, bars or form while a

song plays restarts it with the new recipe. Drums started by hand in Instruments while a song plays come in on its next downbeat and appear as the drums lane here, with their level.

Ask Oscilla: arrange_song, compose_line, song_bounce_arc

## Change one section

One section of the Song Map changes by its id or code — its name, code, bars, description or energy — or a section is inserted or removed; the panel's 'last change' line is written from the difference, for example 'Breakdown 25–32 → 25–30'. The same rules as setting the map apply.

Ask Oscilla: song_map_update, song_map_set, song_map_get

## Compose line

Ask Oscilla to compose a bass, lead, pluck, pad, arp or chord row against a key and progression. A seed repeats the same line. With harmony rules — the rules in force, or rules by name — a lead or pluck line takes the melody's habits (leaps, high at both ends, ending a step under its start, never on the root, within the home field), pads voice from the top with the smallest moves, and the bass has Euclidean grooves among its templates.

Ask Oscilla: compose_line, project_list

## Fill the chord pads

Oscilla can replace the pads with any set that fits — 'darker chords for the breakdown' — naming the key and the chords; numerals and notes are worked out from the names when not given, the first chord is the home pad, and a label says what the set is. You can still press any pad, and Reset returns the key's own layout.

Ask Oscilla: chords_set_pads, chords_get, chords_set_presets

## Harmonize a melody three ways

Chords under a melody, three ways to hear and pick: plain triads, colour (sevenths and ninths) and open (sus chords and fifths), each voiced from the melody down — the melody is every hit's top note, the chord changes where the bass moves and at every bar line, the bass is always a chord tone, nothing sits a semitone under the top, and the rules filter the chords. The melody comes as notes or from a MIDI file; each option is written as a MIDI file under Documents ▸ AudioCortex ▸ MIDI ▸ Harmony with melody, harmony and bass tracks, never over an earlier one (a second take of the same melody gets the next number), so midi_perform can play it through the app's instruments while the Tracks room plays the song.

Ask Oscilla: harmonize, midi_perform

## Hide the panel

Hides the Song Map panel (the chords stay).

Ask Oscilla: chords_hide, chords_show

## Hide the Song Map

Folds the Song Map's section in the Oscilla pane away; the map and the notes are kept, and song_map_show brings it back.

Ask Oscilla: song_map_hide, song_map_show

## Job status

Ask Oscilla how a long record is going. A record that takes longer than a tool call waits returns a job id; job_status reports running, done or failed, the bar it is at, and the full result once done. Stop ends a running job early and keeps what was played. The reply carries done (true or false) and, once finished, the mix path at the top as well as inside the result.

Ask Oscilla: job_status, song_stop, arrange_song, song_bounce_arc

## Lisp arrangement forms

A form script computes the song’s arrangement as data. Choose a saved form or supply a script through Oscilla. The script sees bars, seed, tempo, key, scale, chords, voices and roles. It returns starting rows, drums, levels and changes at bars. Quote data symbols such as 'rest and 'intro. The language has functions, lists, seeded choices, rhythm and harmony helpers; execution stops at step and depth limits. It has no network or general file I/O. Preview with plan_only before recording.

Ask Oscilla: arrange_song

## Oscilla's inbox

The notes you leave for Oscilla — from the Song Map (a note on a section), the Chords tab (a progression) a note typed on the phone remote, and a Tracks lane (a right-click there: the note carries the lane, the time and the bar, and a pin sits on the lane until the reply) — each with an id, where it came from, what it points at, the words, when, its status and the reply. While a note is new and unread, every tool result ends with one line naming it, so an agent at work sees it on its next call; reading the inbox stops the line. The inbox is saved in the project with the map, and with no project open it waits in the scratch session and moves into the project when one is saved. Settings ▸ AI ▸ “Who answers your notes?” decides: Oscilla in the app takes each note at once, or — with the outside assistant chosen — notes wait here for the agent connected from outside.

Ask Oscilla: inbox, inbox_reply, song_map_get, song_map_set

## Project add track

Ask Oscilla to add an audio layer to a project, with a start offset in seconds to place it on the timeline. App-made files are copied into it; your own source files remain where they are — a file beside the package (an album’s Layers/ next to its Project/) is stored relative, with ../, so the folder moves whole. A project plays up to 64 lanes; past that the request is refused before anything is copied. With warp_to_tempo true the file is first warped to the project's tempo (its beats to the grid, pitch kept) and the warped copy becomes the lane, so a layer locks to the session's BPM. With create on, a title that does not exist yet becomes a new project first. With create true, `bpm` sets the new project's tempo.

Ask Oscilla: project_add_track, project_list

## Project bounce

Ask Oscilla to bounce a project's arrangement. It writes the next numbered render through the project's effects board.

Ask Oscilla: project_bounce

## Project list

Ask Oscilla to list saved projects and their renders.

Ask Oscilla: project_list

### Project open

Open a saved project to restore its instruments, rows, kit and Tracks lanes. Continue from the piece's saved state.

Ask Oscilla: project_list, project_open

### Project save

Ask Oscilla to save the current instruments and kit under a project title. This saves state without performing the piece. A project that is only lanes needs no voices: pass none and add tracks.

Ask Oscilla: project_save

### Projects and saved versions

A project is a .acproject package under Documents/AudioCortex/Projects: the take's sounds, rows, arc, plan, drum kit, the recorded stems, and every bounce as a numbered render — plain JSON and real files. One is made by Record in the Arranger (each recorded take becomes a project, named from the recipe) or by asking Oscilla to save (project_save). Projects… in the Arranger's transport, or File ▸ Open Project…, puts one back on the instruments (project_open does the same). Save after changing sounds; add tracks to extend it; bounce to write another numbered render. App-made audio is copied into the package; your own source files are referenced in place, so keep them available. A copied file is stored relative to the package, so the package can be moved or renamed and its stems come with it.

Ask Oscilla: project_add_track, project_bounce, project_list, project_open, project_save

### Read the Song Map and what you point at

Reading the Song Map returns every section with its bars and times, the section you have clicked — what you mean by 'this part' — and your pending notes. Reading marks the notes seen, so the line that rides on tool results stops. While the map is showing, Oscilla checks it at the start of a turn.

Ask Oscilla: song_map_get, song_map_set, inbox, inbox_reply

### Set the Song Map

The Song Map is a section of the Oscilla pane with the song's sections as a strip and a list: a code, a name, bars, times and a line each, so you and Oscilla point at the same thing. Oscilla names the sections by what they are in this song, never from a template. song_map_set gives it the title, BPM, key, length and the sections; times are computed from the bars, the BPM and the meter. Sections must be in order, must not overlap, and keep their own codes; a set that breaks a rule is refused naming the clash and the map stays as it was. The map is saved in the open project (or the scratch session until one is saved).

Ask Oscilla: inbox, song_map_get, song_map_set, song_map_show, song_map_update, song_map_working_on

### Show or hide the Song Map

song_map_show opens the Song Map inside the Oscilla pane — the pane comes out if it was folded — with the song's sections as a strip and a list; song_map_hide folds the map's section

away and the map is kept. You open and close it yourself with the map button at the top of the Oscilla pane or Window ▸ Song Map; the agent brings it back. It is not a window of its own: the map is for work with an agent, so it lives where the agent lives, and people not using agents never meet it. open_window {window: 'songmap'} is the same.

Ask Oscilla: song_map_show, song_map_hide, song_map_set, open_window

### Show the Chords tab

Opens the Song Map panel on its Chords tab; chords_hide hides the panel. The pads are the key's chords as the harmony rules allow them (Rainawave softens the major ones), and a row of next pads appears after a pad is pressed.

Ask Oscilla: chords_show, chords_hide, song_map_show

### Song bounce

Ask Oscilla to record a running performance for a duration. It saves stems and a mixed WAV in Recordings. A bounce joins the bounces already in Tracks; `listen` makes the files without touching Tracks. `seconds` is optional (8 when not given); `voices` is required.

Ask Oscilla: project_list, song_bounce

### Song bounce arc

Ask Oscilla to record a section with changes at specified bars. Drum patterns, rows and parameters change as the section develops. A long section comes back as a job that finishes on its own; job_status follows it. Stopping a record early keeps what was played and puts the performance back as it was before the section. `background` true returns a job id at once; poll job_status.

Ask Oscilla: project_list, song_bounce_arc, job_status

### Song start

Ask Oscilla to start named instrument rows together at a tempo. The voices and optional drums share a downbeat.

Ask Oscilla: project_list, song_start

### Song stop

Ask Oscilla to release the named performance voices and stop the drum sequencer.

Ask Oscilla: song_stop, stop_all

### Suggest a progression

Oscilla writes a progression into the steps for you to hear and edit — an answer to a note, or an idea — with a name to save it in the project and the section it is for. Rests are '' or 'rest'. Steps you recorded and have not sent are yours: the tool refuses to write over them unless replace is true. Her own pads stay when the key is given; note names follow the key — sharp keys spell with sharps, flat keys with flats.

Ask Oscilla: chords_set_progression, chords_get, inbox_reply

### Synth sequence

Ask Oscilla to set a melody row on a voice. Steps are sixteenth notes; -1 rests and -2 holds.

Setting a row does not start playback.

Ask Oscilla: project_list, synth_sequence

## The Chords tab

The Chords tab of the Song Map is a chord machine for notating: hex pads with the key's home chord in the center, the other six chords of the key round it and color chords outside, each labeled with its name and its numeral; pressing one plays it on the keyboard's sound. Under them a strip of steps like the drum machine's — eight, + for more up to 32 — that you record a progression into: with Record on, a pad writes its chord into the step under the cursor and the cursor moves on; a step is tapped to overwrite it; Rest and Clear; the same chord in two steps in a row is a rough 'this lasts longer'. Steps have no tempo. Play hears the row at an even pace. Send to Oscilla puts the progression in the inbox — the key and the chords in order, nothing about timing — on the selected section if any; Oscilla decides timing and voicing. Presets load a row; Oscilla can fill the pads and add presets. The row and the saved progressions live in the project with the map. The rules in force and the next pads come back too.

Ask Oscilla: chords_get, chords_set_pads, chords_set_progression, chords_set_presets, chords_show, inbox, song_map_get

## The harmony rules

The rules in force for harmony — which chords are allowed and liked, and the melody's habits — read from a rules file. Two come with the app: plain pop (every diatonic chord, major triads and dominant sevenths included) and Rainawave (no plain major chord, a major chord only softened, fifths always fine, no leading tone, melodies that leap, sit high at both ends, end a step under their start and never land on the root). A file of your own in Documents ▸ AudioCortex ▸ Harmony is read before the copy in the app, so the tracker and this app can read one file. The Chords tab's Rules menu chooses the rules in force; Oscilla reads them here and can choose for you.

Ask Oscilla: harmony_rules

## The note two parts disagree on

Where two parts rub a semitone or a minor ninth while both sound — lock and key — by bar, with both notes named, from lines given as notes or from every melodic track of a MIDI file. The notes outside the key are listed too. One such rub can be the one dark device a passage wants; two or more are a mistake to find.

Ask Oscilla: clash_check

## The pads’ sound (tool)

The Chords tab’s pads can play three kinds of sound: the keyboard’s own sound as its knobs are set (the default), the sound selected in the Library as a sampler on the same root the keys use, or any built-in patch from the synth bank — Nova, Aurora or Mystical. A patch or the selection plays on the tab’s own six voices, so the keyboard’s pool is left as you set it. The choice is kept with the song’s chords. On the tab it is the Sound menu beside the key; Oscilla sets it by name.

Ask Oscilla: chords_sound, chords_get, synth_preset, chords_set_pads, chords_set_progression, chords_set_presets

## What chord comes next

The chords that could follow one — after Am, say — ranked the way the tracker ranks them: by root motion (a fourth up pulls hardest, a step down is the minor-pop walk) and the rules' leanings, the best of each root first, each voiced in three notes to hear. A melody note keeps that note on top and drops the chords it rubs against. The suggestions also land on the Chords tab as a row of next pads to tap and hear — the same row the tab shows after a pad is pressed or the cursor moves.

Ask Oscilla: chords_suggest

## What Oscilla is working on

The amber line at the top of the Song Map: one sentence on what Oscilla is doing now, and the section it is in, which is highlighted in the strip and the list. An empty text clears it. Kept current, it lets you follow from the terminal.

Ask Oscilla: song_map_working_on, song_map_set, song_map_get

# BPM Tap

## BPM Tap

Tap along with the music and read its tempo. Press the pad on every beat — it acts the moment it is pressed — or press Space. AVERAGE is the tempo of the whole run of taps; INSTANT is the last two taps alone. Under them: how many taps, how far they wander (±), and whether the run is steady. The light beside them keeps the beat you tapped, so you can hold it against the music. ½ and ×2 correct a hand that tapped eighth or half notes. Click sounds a click on every tap. A pause of two and a half seconds starts a new run; Reset clears it. The delay times of the average tempo are listed under the pad. Set Song Tempo gives it to the Tracks room — its tempo field, grid and metronome; nothing changes until you press it.

Ask Oscilla: detect_bpm, tracks_tempo

# Delay Finder

## Delay Finder

Enter a tempo in BPM to see note-length delay times in milliseconds and their rates in Hz. Copy the value to the effect you want to set. The calculator does not change an effect.

# DJ

Read dj_status before loading, syncing or crossfading.

## DJ

DJ has two decks, a crossfader and a crate for each deck. Drop sounds onto decks or into crates — or onto the DJ item in the sidebar: the first empty deck takes one, the rest go to deck A's crate. Each deck reads BPM (the tempo it plays at now, following the tempo slider) with the original BPM under it, the musical key with the bar under it, and elapsed and remaining time; the band around the platter is the position in the track and flashes for the last twenty seconds. PLAY plays the track; click it again to pause. CUE is the temporary cue, at the start of the file until you set it: while paused, press CUE to set the cue where the playhead stands; hold CUE while paused to play from the cue, and release to pause and return to it; press PLAY while holding CUE and playback carries on when you let both go; press CUE while the track plays to return to the cue and pause. SYNC matches the deck's BPM to the other deck and, when both are playing, puts its beats on the other's; the deck then stays in sync (SYNC lit) until OFF, which returns it to its own tempo, until you move its tempo slider, or until you load another track. A deck already at the other's tempo comes in on the beat when you press PLAY. The TEMPO slider beside each deck changes its speed by up to eight percent either way, the standard tempo range; the pitch under it is the percentage from the original tempo. Shift drags fine and ⌘-click returns it to zero. Speed changes pitch: there is no key lock. VOLUME is the deck's upfader, its level before the crossfader. HI, MID and LOW are the deck's EQ and FILTER its filter; a vertical drag turns a knob (Shift for fine), and double-click or ⌘-click resets it. The crossfader blends the two decks; Shift drags fine and ⌘-click centres it. The main waveform is both decks' real audio scrolling under a fixed playhead in the center: what is coming is on the right, beats and bar lines are marked, and the bright part of each moment burns hotter. Every deck reads tempo on one scale (78 to 156 BPM, so 174 reads as 87), to a hundredth of a BPM where the track holds a steady tempo. Double-click a sound in a crate to load it, or LOAD NEXT TRACK loads the crate's next; an empty deck's waveform lane says how to load one. Bars are counted from the first beat in the file, including one that falls at its very start. When the room is narrower than the two decks want — the Oscilla pane open, or Text size at Largest — the pair shrinks together to fit, every control still in reach.

Ask Oscilla: dj_status

## DJ break

Ask Oscilla to add a drum break locked to the audible deck. It starts on the requested grid boundary; stop ends the break.

Ask Oscilla: dj_break, dj_status

## DJ crate

Drop sounds in a deck's crate, then use LOAD NEXT TRACK. Each crate belongs to its deck and is saved across launches.

Ask Oscilla: dj_crate, dj_status

## DJ crate read

Ask Oscilla to analyze a crate before playing a set. It reports tempo and key relationships and useful pairs.

Ask Oscilla: dj_crate_read, dj_status

### DJ crossfade

Move the DJ crossfader from A at the left to B at the right. Ask Oscilla for a timed fade to make the move gradually.

Ask Oscilla: dj_crossfade, dj_status

### DJ drop

Ask Oscilla to place a one-shot over the decks. It can land on a beat, bar, phrase or immediately, with optional deck ducking.

Ask Oscilla: dj_drop, dj_status

### DJ eq

Use a deck's low, mid, high and filter controls. The center is flat; reset restores every band.

Ask Oscilla: dj_eq, dj_status

### DJ load

Drop a sound on DJ deck A or B, or ask Oscilla to load it. Playback is available while tempo and key analysis finishes.

Ask Oscilla: dj_load, dj_status

### DJ play

Use PLAY, CUE and SET on a deck, or ask Oscilla to play, pause, stop or set its cue point.

Ask Oscilla: dj_play, dj_status

### DJ status

Ask Oscilla to inspect both decks before mixing. Each deck’s key is given with its relative beside it (C major also reads as A minor), and a rolling take is shown.

Ask Oscilla: dj_status

### DJ suggest

Ask Oscilla what from the current Library list would mix with a deck. Candidates are ranked by tempo and harmonic compatibility.

Ask Oscilla: dj_status, dj_suggest

### DJ sync

Use SYNC to pull a deck's tempo to the other deck's; when both are playing it also puts the deck's beats on the other's, and a deck already at the other's tempo comes in on the beat when played. This changes pitch along with speed. The deck then reads synced until OFF (off: true — back to its own tempo), a hand on its tempo slider, or a load.

Ask Oscilla: dj_status, dj_sync

### DJ tags: tempo, key, energy

Ask Oscilla to tag files the way a DJ program shows them: the tempo, the key and its position

on the Camelot wheel (8A, 11B — neighbors mix), and an energy figure from 1 to 10. The energy is the ear's measure — loudness carries seven tenths of it and how many hits a second the rest — not a model's opinion. Tag a deck's crate, a set list, or any paths; the exports use the same tags.

Ask Oscilla: dj_tag, dj_export, dj_crate, detect_key

## Export a playlist to the decks

Ask Oscilla to send a playlist to the decks people play on: rekordbox takes an XML (File, Import Collection in XML format, or the bridge file in its Advanced preferences); Traktor takes an NML dragged onto its playlists; Serato takes a crate file copied into the Serato folder's Subcrates on the drive the files live on. The tracks are a deck's crate or any paths; every one is tagged first with its tempo, Camelot key and energy, and those ride in the file. The export lands in the DJ folder under Exports in the AudioCortex documents, named after the playlist, unless another folder is given.

Ask Oscilla: dj_export, dj_tag, dj_crate

## Record the decks

Ask Oscilla to record the decks: start before the mix, stop after it, and the file — both decks, the crossfade and the break drums as heard — lands in Recordings with its length and level, ready for Tracks or a judge. Status says whether a take is rolling, and the DJ status shows it too. If part of the take could not be written (a full disk), stop says so and gives the length that was.

Ask Oscilla: dj_record

# Instruments

Choose the voice before synth_preset, synth_param or drum_pattern.

### Drum bounce

Bounce the drum machine's rhythm to a sound file. drum_bounce renders the pattern as it stands — the grid, its kit, tempo, swing, accents and master effects — for `bars` bars (default 4, at most 64) plus a short tail, to a stereo WAV in Documents/AudioCortex/Recordings, and says where. It is the same as the Instruments room's Bounce menu (4, 8 or 16 bars), which also shows the file in the Finder. Oscilla can also bounce a whole song's drums in one go — verse, riser, chorus with a fill, a break — each section its own pattern for its bars, one after another on the bar line.

Ask Oscilla: drum_bounce, drum_pattern, drum_kit

### Drum hit

Ask Oscilla to trigger a kit voice such as kick, snare or closed hat.

Ask Oscilla: drum_hit

### Drum kit

Choose the built-in sampled kit or synthesized drums, or one of the eight kits the app ships — Classic Pop, Industrial, Hybrid Boxes, Arcade, Spectral, The Ice Cube, Refined, Tape Dust, Cathedral — each made from the drum machine’s own synthesis and carrying a starter pattern. Ask Oscilla to save and load named kits with their settings and patterns; a shipped kit cannot be deleted, and saving under its name makes your own copy. A SoundFont drum kit loads too: Oscilla puts its sounds on the ten voices by the usual drum keys (kick on 36, snare 38, hats 42 and 46…), and saving keeps it as a kit.

Ask Oscilla: drum_kit

### Drum pattern

Ask Oscilla for a drum pattern. Each voice has sixteen steps: x hits and a dot rests. Setting the pattern starts the sequencer. The kit's voices are kick, snare, hat-closed, hat-open, clap, tom-lo, tom-mid, tom-hi, cowbell and clave; everyday names such as hat, hi-hat, open hat, bass drum and rim are understood. A row of one, two, four or eight steps loops to fill the bar; other short rows are padded with rests, and the reply says which.

Ask Oscilla: drum_pattern

### Drum sample

Assign a file to a drum voice. Its tuning then changes sample playback speed. Clear the sample to restore synthesis.

Ask Oscilla: drum_sample

### Drum stop

Stop the drum sequencer. This does not stop every other sound source.

Ask Oscilla: drum_stop

## Drum tune

Ask Oscilla to change a drum voice's tuning, level, decay, pan or color.

Ask Oscilla: drum_tune

## FM bank

Ask Oscilla to inspect a six-operator FM SysEx bank or single-voice dump before choosing a voice.

Ask Oscilla: fm_bank

## FM patch

Ask Oscilla to load a six-operator FM voice from a SysEx file by index or name. The translated sound uses the app's FM core.

Ask Oscilla: fm_patch

## FM preset

Browse FM presets in Sounds or ask Oscilla to filter by name, category or words. A unique match can be applied to a voice. Ask loosely — a glass bell, an electric piano — and the closest preset is applied; the reply names a few others that matched. Names are shown cleaned up: decoration, stray symbols and shouting capitals from the original banks are tidied, and a voice whose stored name is only a signature takes its bank's name and number. The bank files are not changed, and a preset can still be asked for by its original name.

Ask Oscilla: fm_preset

## Grow patch

Ask Oscilla to evolve a synth toward a known sound category. It renders candidates and keeps improvements; the winning patch remains on the voice.

Ask Oscilla: grow_patch, model_categories

## Instruments

Play and shape the synth and drum machine here. Choose a sound, use the keyboard, and change oscillator, filter and envelope controls. Envelope times are seconds; filter cutoff is Hz. Arp: switch it on and hold keys — one voice plays the held notes one after another (the lowest is the root) in time with the Tracks tempo; 1/8, 1/16 or 1/32; Up, Down, Up-Down or Random; Octaves repeats the pattern higher; Gate sets how long each step sounds. Named arranger voices and the panel instrument are separate targets, so tell Oscilla which voice to change. The drum grid follows the machine: a pattern Oscilla makes lights its steps here, with the tempo, the Run button and a playhead on the step that is sounding. Space starts and stops the drum machine, as Run does, whenever no text field is being typed in. Above the synth, a Playing row names the voices Oscilla has going and the sound on each. Leaving the room does not stop the music; use Stop. The Playing row shows the parts that are sounding now (a chord's voices as one part), not every voice the session has made. The synth's knobs edit the keyboard's own synth — and any part whose sequence is playing — until a part in the Playing row is clicked; then they show that part's values and shape it while it plays, and the header says which. Click the part again, or the button in the header, to return to the keyboard's synth. The green dot on Instruments shows whenever its drum machine or synth voices sound, a song's included. Changing kits keeps the pattern that is playing. Run, pressed while a song

plays in the Arranger, starts the beat on the song's next downbeat at its tempo, and the Arranger shows it as a drums lane with a level. The Playing row's parts are hex-cut chips under a PLAYING label: the chosen one sits on the sound well, and a part's dot is amber while it sounds. Bounce (4, 8 or 16 bars) writes the drum machine's pattern as a WAV to Documents/AudioCortex/Recordings and shows it in the Finder. Drop a SoundFont (.sf2) or a folder of notes on the room and the synth plays it: the keys reach every zone. On a Bluetooth output the note slot under the grid says, when Run starts, that what you hear is about a quarter second behind the beat and the lit step; wired headphones, the Mac's speakers or an interface keep them together. Right-click a synth knob for MIDI Learn: move a control on your MIDI controller and it turns that knob, and the knob on screen follows.

Ask Oscilla: drum_pattern, synth_param, synth_preset

## Pd control

After loading a Pure Data patch, ask Oscilla to send knob values, notes, floats or bangs to it.

Ask Oscilla: pd_control, pd_load

## Pd load

Ask Oscilla to load a Pure Data patch. One patch runs at a time; unsupported external objects can prevent loading.

Ask Oscilla: pd_load

## Pd record

Ask Oscilla to record the running Pure Data patch into Recordings. Controls can change while it records. If part of the take could not be written, the reply says so.

Ask Oscilla: pd_load, pd_record

## Pd stop

Close the loaded Pure Data patch and silence that engine.

Ask Oscilla: pd_stop

## Poly voice

Ask Oscilla to make a named voice polyphonic before assigning chords. Two to eight sub-voices share the patch and bounce as one stem. The MIDI keyboard's own voice (midi-keys) is one voice by nature and is refused.

Ask Oscilla: poly_voice

## Synth note

Ask Oscilla to play a note on a named synth voice. It sounds immediately; use a render to keep it.

Ask Oscilla: synth_note

## Synth param

Change a synth parameter through Oscilla. Cutoff uses Hz, envelope times use seconds, and pan runs from left at -1 to right at 1. sample-root may be the word auto: the root is then read from the loaded sample so it plays in tune. A SoundFont (.sf2) or a folder of notes loaded as

the sample makes the synth an instrument of zones: each note plays its own sample at its key and velocity, with its root and loop — on the keyboard, in sequences and under the Knob Panel.

Ask Oscilla: synth_param

## Synth preset

Choose a Nova, Aurora or Mystical patch in Sounds, or ask Oscilla to apply one to a voice. A clean baseline is applied before the patch, so earlier settings do not leak in. A loose description — acid, warm pad, sub bass, piano — picks the nearest patch and says which; a bare “bell” goes to the FM bells, while the Mystical bells — real ones, recorded — are asked for by their full names. Shipped patches: Mystical Tibetan Bell, Mystical High Bell, Mystical Tuu Bell, Mystical Small Bell.

Ask Oscilla: synth_preset

## Synth render

Ask Oscilla to render a note from the shared synth. It records the live engine to a file you can play or reuse.

Ask Oscilla: synth_render

## Wave program

Ask Oscilla to program a synth's frame-by-frame timbre sequence. Empty byte arrays clear it.

Ask Oscilla: wave_program

## Wavetable

Ask Oscilla to load a wave file or folder onto a voice. Single cycles become frames; sustained samples supply a detected period. Position moves across the table.

Ask Oscilla: wavetable

# Library

Read the current list before searching the disk.

## A loop at the session's tempo

Ask Oscilla for a loop rendered at a tempo so it lands in a DAW session on the beat: the beats are found and warped to the grid with the pitch kept — the warp tool at its simplest. The same thing happens when a person Option-drags a sound out of the Library with a session tempo set in the Metadata window: the dropped file is the loop at that tempo.

Ask Oscilla: render_at_tempo, warp

## Activate AudioCortex

Everyone gets 14 days: the first launch says the trial has begun and when it ends, one note two days before the end says so and suggests three things to try, and after the fourteenth day the app opens only to this window. Once on each Mac, it offers Need more time? One more week — seven more days, with nothing to fill in; after that it waits for a code. A license of either kind ends the trial's say. In the activation window, paste the code from your purchase email and press Activate. The window reports success or the reason activation failed. Paste, Purchase and Request Code provide the related actions. An activated license is checked locally on this Mac. Deactivate This Mac releases this installation through its confirmation flow. Assistant sign-in and assistant keys are separate from the app license.

## Add to a scan set

Ask Oscilla to put sounds or folders into a named scan set (made if it does not exist yet), or to remove a set. Folders are scanned recursively; single sounds are members in their own right. load_scan_set shows the set.

Ask Oscilla: scan_set_add, list_scan_sets, load_scan_set

## Add to the Crate

Ask Oscilla to add files to the Crate. The Crate collects references without moving the files.

Ask Oscilla: bin_add, get_selection, list_files

## An index beside the library

A library on a share is slow to search from another machine: every folder is a round trip, and a search can time out half way and quietly miss files. An index fixes that. It lists every audio file once — its path relative to the library, its size and date, duration, sample rate and channels, and if asked its tempo and key — in a small file beside the files, so every Mac that mounts the share searches that one file instead of the disk. A rescan reads only what changed. Best made on the machine that holds the files, with no window: AudioCortex --scan-agent followed by the folder, adding --analyze for tempo and key and --watch to keep it fresh as files arrive and once a night.

Ask Oscilla: scan_index, scan_index_status, index_search, find_files

## Analyze

Ask Oscilla to analyze a file for category, frequency, tempo, key and format. Analysis does not change the recording.

Ask Oscilla: analyze, get_selection, list_files

## Ascension

Ask Oscilla for a rising or falling octave-layered glissando from a sound. Cycle controls seconds per octave.

Ask Oscilla: ascension, get_selection, list_files

## Bones

Ask Oscilla to split a sound into sustained harmonic material and percussive or noisy material.

Ask Oscilla: bones, get_selection, list_files

## Browse folder

Ask Oscilla to open a folder in Library. Its audio files become the current list.

Ask Oscilla: browse_folder, get_selection, list_files

## Chromagram: notes over time

Twelve pitch classes across time, color = energy, with the best chord per bar written above the grid (at bpm from offset), the key and the tuning offset in cents. compare stacks a second file on the same grid — a lane against the mix, a layer against the stems — and lists the bars where their chords disagree.

Ask Oscilla: chromagram_image

## Classify

Ask Oscilla to classify a file. It reports categories and confidence; these are model judgments, not labels you must accept.

Ask Oscilla: classify, get_selection, list_files

## Coming from AudioFinder

AudioCortex 2 is the app that follows AudioFinder. AudioFinder 6.1 still runs on today's macOS, and its AudioCortex menu — Open in AudioCortex 2 — hands the selected sounds to this app, which opens the Library at their folder with them selected (Space plays); nothing is built into AudioFinder any more. Where things went: the Device Browser and Finder views are the Library's column browser, which starts at the top of this Mac; Saved Lists are collections; the Bin is the Crate (right-click a sound ▸ Add to Crate; View ▸ Clear Crate empties it); the Sample Tool is the Editor — right-click a sound ▸ Open in Editor — with the zoom-and-scroll bar under its waveform; Guess Key and Tap are the Tone Finder and BPM Tap rooms, and a sound's analysis carries the same readings; AI Classify is Analyze, done on this Mac with no helper, and the title-bar search begins with ? to find sounds by how they read; a MIDI keyboard plays the selected sound as it did; SynchroFinder is Finder Selection, under the Window menu — a panel that follows what you select in the Finder and plays it. Your tags come over too: Window ▸ Tags… ▸ Import from AudioFinder brings the tag list and the tags on your files, once. The Metadata window and tempo-synced export into a DAW are not here yet. Tracks, DJ, Nodes, the Arranger, Instruments and Oscilla are new. Licenses: an AudioFinder bought in 2020 or later includes AudioCortex 2 — press Request Code in the activation window with that purchase; everyone else has the 14-day trial, and no subscription, ever.

Ask Oscilla: analyze, detect_key, detect_bpm, file_metadata, create_collection, bin_add, tags

## Control AudioCortex from your phone

Settings ▸ Phone turns on the phone remote. Scan the code it shows with your phone's camera and open the link: the phone shows the transport (play, stop, record, loop, the metronome), the Tracks lanes with their faders, mute, solo and record-arm, the Knob Panel's knobs, and a box for a note to Oscilla — at home or anywhere else, with nothing to install on the phone and no router or VPN setup. Your Mac and your phone meet through Iced Audio's relay; what passes between them is encrypted with a key only the two of them have, kept in the code you scanned. The phone can only play and mix — it never reaches your files or settings. Add the page to the phone's Home Screen to keep it one tap away. While a phone is connected, Settings ▸ Phone says so. New pairing makes a fresh code, and phones paired before stop working until they scan it. The Sing button records you singing or humming an idea on the phone — in the car, anywhere — and sends it to your Mac, which writes the melody down as notes and leaves it for your assistant as a note; the recording is kept in Documents/AudioCortex/Recordings. Every press is acknowledged and written to the history your assistant reads (as coming from the remote), so nothing you do on the phone is lost or invisible; a press repeated after a dropped connection is applied once. The Song Map panel shows what the assistant is working on and has a Checkpoint button that keeps the song as it is now.

## Convert files

Ask Oscilla to convert sounds to another format — WAV, AIFF, CAF, FLAC, Apple Lossless, AAC or MP3 — and, if you like, another sample rate, bit depth or channel count; the Library's DSP ▸ Convert… does the same by hand, and DSP ▸ Convert To takes a format with its usual settings. A drop to 16 bits or fewer is dithered unless you say not to. MP3 and AAC take their real settings: variable bit rate with a quality from V0 (best) to V9 (smallest), or a constant or average bit rate in kbit/s. The files you give are never touched: each result is written next to its source or into the Process output folder (or a folder you name), under a new name, never over a file already there, and a comment carried in a WAV or AIFF travels to a WAV or AIFF result. A file that cannot be converted is named in the reply and the rest go on; a long batch can be stopped with render_cancel.

Ask Oscilla: convert_files, render_cancel

## Create collection

Ask Oscilla to save selected paths as a named collection. It appears under Lists in the sidebar.

Ask Oscilla: create_collection

## Descriptors

Ask Oscilla for browsing measurements such as brightness, noisiness and harmonicity. Tempo and key are optional because they take longer. A big folder is measured to a time budget; the reply says how many files remain, and asking again continues where it stopped because finished files are cached. Files still in iCloud and not on this Mac are counted and requested.

Ask Oscilla: descriptors, get_selection, list_files

## Detect BPM

Ask Oscilla to detect the tempo of an audio file. The result is an estimate from its beats. The tempo is the same settled estimate analyze_mix and judge_musical give, so the three agree on the octave; the raw beat detector's figure is reported beside it when it differs.

Ask Oscilla: analyze_mix, detect_bpm, get_selection, judge_musical, list_files

## Detect key

Ask Oscilla to estimate an audio file's key. It analyzes pitch energy; it does not transcribe every note.

Ask Oscilla: detect_key, get_selection, list_files

## Embedded metadata (AudioFinder's)

Ask Oscilla what a sound says about itself in its own file: the metadata AudioFinder embeds — Creator, BPM, Beats, Time signature, Key, Library, Genre, Category, Rating, Tags, Notes — kept in a chunk inside a WAV or AIFF, so it travels with the file. Sounds tagged in AudioFinder since 2007 read here, and sounds tagged here read in AudioFinder 6. Other formats carry none.

Ask Oscilla: metadata_read

## Envelope

Ask Oscilla to inspect loudness over time. The curve shows transients and pumping, normalized to its peak. Long files are decimated to at most a few hundred points (keeping each bin's peak); `points` sets the cap. The file is read in pieces, so a thirty-minute bounce is as welcome as a one-shot.

Ask Oscilla: envelope, get_selection, list_files

## File metadata

Ask Oscilla to inspect a file's format, slice markers and loop points, the WAV's comment, and the marks the editor keeps beside the file: bookmarks, the chops dragged out of it, and for a chop the sound it was cut from.

Ask Oscilla: file_metadata, get_selection, list_files

## Find by category

Ask Oscilla to search a folder for sounds by category. Results depend on the classifier's vocabulary.

Ask Oscilla: find_by_category, get_selection, list_files

## Find duplicates

Ask Oscilla to search for duplicate sounds. Inspect the reported groups before deciding what to keep.

Ask Oscilla: find_duplicates

## Find files

Ask Oscilla to search a folder and its subfolders by name. Results include paths and the subfolders with the most matches. When the folder has an index beside it (scan_index, or the scan agent on the machine that holds the files), the search reads the index instead of the disk — instant over a share — and says how old the index is.

Ask Oscilla: find_files, scan_index

## Find similar

Ask Oscilla to find sounds resembling a reference file. It compares sound-category vectors.

Ask Oscilla: find_similar, get_selection, list_files

## Find sounds on the web

Ask Oscilla to look for openly licensed sounds on the web when the Library has none — whispers, breaths, voices, field recordings. It searches Openverse (Freesound, Wikimedia Commons, Jamendo and more) and lists title, creator, license, length and a link for each. CC0 is the default, which owes nothing; the Creative Commons variants owe attribution, and the import writes it into the file.

Ask Oscilla: search_sounds_web

## Finder Selection

Finder Selection is a small floating panel that follows what you select in the Finder and plays it — AudioFinder's SynchroFinder, for AudioCortex 2. Open it from Window ▸ Finder Selection; it is a helper of its own, so it runs light and goes on without the rest of AudioCortex. Click a sound in the Finder and the panel shows its name, its format, rate, channels and length, and its waveform; with AUTO on it plays at once, and a click on the waveform plays from there. Loop, Stop and Play are the hexes at the left; the volume is the slider at the right; AudioCortex opens the sound in this app. Its icon sits in the menu bar too, with the same choices and Quit. Window ▸ Open Finder Selection at Login starts it with your Mac — no prompt; Settings ▸ General ▸ Login Items & Extensions lists it. The first time it looks at the Finder, macOS asks once whether "AudioCortex Finder" may control the Finder: Allow. If it was refused, the panel says so and opens Privacy & Security ▸ Automation for you. It runs on this Mac's AudioCortex 2 license: activate here once and the panel follows; with no license it asks you to open AudioCortex 2. It plays what macOS can read — WAV, AIFF, CAF, FLAC, MP3, M4A and the rest — and names anything else as not a sound.

Ask Oscilla: open_window

## Freeze

Ask Oscilla to sustain one moment's spectrum. Choose the time, or let the app choose the loudest moment.

Ask Oscilla: freeze, get_selection, list_files

## Get annotation

Ask Oscilla to read the rating and comment attached to a file.

Ask Oscilla: get_annotation, get_selection, list_files

## Get selection

Select files in the Library list, docked or popped out, then ask Oscilla to work with them. This reads the selected paths; if a redraw has just cleared the list's highlight, it reads the files you last picked there, as long as they are still listed.

Ask Oscilla: get_selection

## Glossary: list, collection, scan set, crate, analyze, classify, deep scan, MIDI index

List — what the Library's table shows right now: a folder's sounds, a collection, a scan set, a search's results. Collection — a saved list of sounds you chose, kept in the sidebar; + makes

one, drag sounds in, a collection stays when the folders change. Scan set — a named set of folders that are scanned recursively (and single sounds you add): in the sidebar + ▸ New Scan Set, or ask Oscilla (scan_set_add); open it from the sidebar and it lists everything under those folders. Crate — the session's holding area (right-click ▸ Add to Crate; the sidebar's Crate badge counts it): sounds you set aside while you work, to send to Tracks, DJ, the Explorer or Nodes together or to export; Clear Crate empties it. Analyze — reading a sound: tempo, key, level and its category; a folder analyses as it lists, and the Analyze button runs it on what is listed; the results are remembered so nothing is read twice. Classify — the part of analysis that names what a sound is, one of 454 categories (kick, pad, vocal…) with a confidence, by the sound model in the app. Deep scan — walking a whole folder tree in the background, listing every sound under it; the Deep Scan button starts it, it can be paused, and what it found is kept for next time; the * search then finds any of it by name or by sound. MIDI index — the catalog of what is IN your MIDI files (key, tempo, meter, groove, roles), built on this Mac in seconds, which the MIDI search in the MIDI room and Oscilla's MIDI tools read. REX files (.rex, .rx2, .rcy) are listed as sound files but their slices are not decoded: they play only where the system can read them; export them as WAV from the program that made them to use them here.

Ask Oscilla: scan_set_add, load_scan_set, list_scan_sets, analyze, classify

## How fresh an index is

Whether a folder, or one above it, has an index: when and where it was made, how many files, how old it is and whether it is stale; with check, how many of its files changed or went since.

Ask Oscilla: scan_index_status

## Import a sound from the web

Ask Oscilla to bring one sound from a web address into the Library. It lands as a WAV in Documents ▸ AudioCortex ▸ Imports, in the folder you name, and the license, creator, title and source are written into the file itself and into its Finder comment, so where it came from travels with it. Mp3, wav, aiff, flac and m4a decode, and ogg (Vorbis) from macOS 15.4; opus does not.

Ask Oscilla: import_url

## Import raw PCM

Ask Oscilla to bring a raw PCM file into the Library as a WAV. Music Thing Radio Music banks are .raw files — 16-bit mono at 44.1 kHz, no header — which is the default here; say the sample rate, channels, bit depth and byte order when they differ. The WAV lands in the Imports folder of the AudioCortex documents, in a subfolder of your choosing (Raw by default), named after the raw file unless you name it.

Ask Oscilla: import_raw, import_url

## Library

Browse folders, collections and scan sets here. The column browser starts at the top of this Mac; the breadcrumb above it shows the folder you are in and climbs back up when clicked. Click a file to audition it. The preview under the list shows one scope at a time, chosen from the pop-up on its bar: the sound as a Spectrogram or a Waveform, or one of Oscilla's own pictures of it — Chroma (notes over time, the chord per bar, the key and tuning), Loudness (short-term and momentary LUFS, true peak, crest, sections), Stereo (width and correlation, the goniometer, a low-end mono check) and Balance (the average spectrum over a reference

track you pick with Reference…) — drawn by the same tools an agent calls (chromagram_image, loudness_timeline_image, stereo_field_image, tonal_balance_compare), so what you see is what it reports; a playhead rides the player over the time pictures. Changing the scope never stops a sound that is playing. The scope is kept across launches. Right-click for Rename… (one file, in place; ratings and comments follow), Batch Rename… and the rest. Drag sounds onto the Tracks, DJ, the Explorer or Nodes item in the sidebar to send them there. The title-bar search filters names, categories and embedded descriptions. Begin it with ? to describe a sound instead ("? deep bass"): the words are matched to the sound categories and the analyzed sounds are ranked by how much they read as those categories — on this Mac, no helper; sounds never analyzed can't rank. Begin it with * to search everywhere — every analyzed sound on this Mac, not the list in front ("* kick", "*? deep bass"); the results come as a list of their own. Help has its own search. In the sidebar a green dot marks any room that is making sound — a sound auditioned here lights the Library's own — so you can tell where music is coming from after you leave its room; a blue dot marks a room that has its own window open. At the foot of the sidebar, Recordings opens the folder where the app saves what it makes outside a song, with a count of the files added since launch; its first click explains it, and right-click opens its own help entry (Recordings, songs and what a song keeps). Oscilla, above it, opens and closes her pane beside the room: she is lit in her own color while it is open, and the bar and the highlight mark the room in front. An empty list says so, and how to fill it. A sound with more than two channels plays folded to stereo, so nothing on channels three and up is lost. With Loop off, View ▸ Play Through the List (⌥⌘→) lets a sound that ends hand over to the next row. The volume slider is a fader: its travel is in decibels, the top half the last 12 dB. Sounds Oscilla brings in from the web land in Documents ▸ AudioCortex ▸ Imports, with their license and source written into each file. A sound Oscilla plays is selected in the list for you, its folder opened if the list was elsewhere. A MIDI keyboard plays the selected sound, pitched by the key — middle C as recorded, the octave above at double speed. A scan set keeps what you drop on it — folders or single sounds — in the sidebar or into its list. The counter beside the play button is the studio one: click it to read min:sec, SMPTE timecode, bars|beats, frames or seconds; right-click it for the frame rate (24, 25, 29.97 drop-frame, 30); its nudge keys step the cursor a frame. Play/Stop is an amber hex and Loop a dark hex lit in the sound color when it is on; both follow the Appearance scheme. A sound's original pitch sits on C4 for every sound (AudioFinder's way); Settings ▸ Audio ▸ MIDI In can put it on the sound's own note instead, so a G3 sample sits on G3. Right-click a voice memo ▸ Melody from This Recording → Oscilla writes a sung or hummed idea down as notes and sends it to Oscilla. The list watches the folder you are browsing: a file added, removed or rewritten there shows up by itself within a second or two, and the column you sorted by — date, name, length — keeps its order for new files and for the next folder you open. Space plays and stops the selected sound, and the up and down arrows move to the next or previous sound, wherever the focus sits in the Library, unless you are typing in a field. Tags of your own go on sounds from the right-click menu (Tags), the sidebar's Tags heading lists the sounds that carry one, and #tag in the search finds a whole tag. While Analyze runs, each row's Sounds Like fills as that sound is heard, and the summary's count moves with it.

Ask Oscilla: chromagram_image, loudness_timeline_image, stereo_field_image, tonal_balance_compare, tags

## Library similar

Ask Oscilla for related sounds from your analyzed library. This uses the app's category space and does not need the helper service.

Ask Oscilla: get_selection, library_similar, list_files

## List files

Ask Oscilla to inspect the current Library list. It can read a page or sample a few rows; it does not need to load every result.

Ask Oscilla: list_files

## List locations

Ask Oscilla to read the sidebar's favorites, recents, history, bin or saved searches.

Ask Oscilla: list_locations

## List scan sets

Ask Oscilla to list your named scan sets and their folders. A set's members are folders (scanned recursively) or single sounds; sounds dropped into the list while a set is showing join the set.

Ask Oscilla: list_scan_sets

## Load scan set

Ask Oscilla to load a named scan set. Its folders are scanned recursively into Library.

Ask Oscilla: list_scan_sets, load_scan_set

## Loudness

Ask Oscilla to measure one file or compare several. Loudness is reported in LUFS; comparisons are relative to the first file. A long or high-rate file is brought down to size for the measurement instead of being refused.

Ask Oscilla: get_selection, list_files, loudness

## Loudness over time

Short-term LUFS (3 s), momentary LUFS (400 ms, faint), true peak per second and the crest factor, drawn over time with section markers — passed in, or found where the level moves by more than 5 LU from the section so far (sections at least 20 s). Per section: mean short-term LUFS, the loudest true peak and the lowest crest, with warnings for a drop of more than 6 LU into a section, a crest under 8 dB, and a true peak over −0.3 dBTP.

Ask Oscilla: loudness_timeline_image

## Lush stretch (a dense, wide cloud from a recording or from the synth)

Ask Oscilla to make a lush cloud. The source can be a recording — a guitar chord, a phrase, a few notes — or nothing at all: name some chords (F#sus2, C#m7, B) or just a key, and a sound, and the app's own synth plays them. Either way the sound is stretched until it dissolves, with the left and right sides scrambled separately so it is very wide, and stacked in many layers an octave apart and slightly detuned so it is dense. Several recordings or chords each last about twenty seconds and melt into the next. Ask for pauses and each chord swells in, holds and sinks away, with a rest before the next — more swells, more air. A plucked or struck sound is strummed again and again so the cloud holds. Point it at a track you love and the cloud is steered toward that track's balance, width, breathing and density — its shape, never its notes — and Oscilla tells you how close it came. Without a reference, darkness and low end can be steered. The result is a new file, beside the source recording or in Recordings; nothing is overwritten.

Ask Oscilla: lush_stretch, process, texture_fingerprint

## Melody from a sung recording

Sing or hum an idea into your phone's voice memos — in the car is fine — and bring the recording to AudioCortex. Right-click it in the Library ▸ Melody from This Recording → Oscilla: the app hears the melody under the noise, writes the notes down on the beat grid at the Tracks tempo, names the key, saves a MIDI file in Documents/AudioCortex/Melodies, and leaves the melody in Oscilla's inbox as a note from you, so Oscilla (or the assistant you connect from outside) can build around it. One voice works best. It only listens — nothing is made from your voice. On the phone remote, Sing does the same from your phone: tap, sing, tap again, and the notes arrive.

Ask Oscilla: melody_from_voice, inbox

## MIDI In

A MIDI keyboard plugged into the Mac plays the app, as it did in AudioFinder — every connected keyboard is listened to, and one plugged in later is picked up by itself. A key plays the room in front of you: in the Library (and any room without its own instrument) it plays the selected sound, pitched by the key — the sound's own note under its own key (the root is read from the file; a sound with no clear pitch sits at middle C), each octave up doubling the speed, each octave down halving it, with the sound's own start and end; in the Instruments room it plays the synth, whatever patch is on it. In the Sounds room it plays the sound selected in the browser, chords included, the same way the on-screen keyboard does. With a drum kit selected, the keys play the kit on the General MIDI drum notes. The MIDI button beside that room's keyboard turns this on and off. Channel 10 plays the drum machine on the General MIDI map (36 kick, 38 snare, 42 closed hat, 46 open hat, 39 clap, 41–50 toms, 56 cowbell, 37 and 75 clave). Velocity is honored and the sustain pedal holds. The keys voice is one voice, so a chord sounds as its last key; files longer than two minutes are not loaded into it. A key, pad or knob you map to a control in Settings ▸ Controllers works that control instead, and every other knob and fader on the device can be mapped there too. Settings ▸ Audio ▸ MIDI In turns it off, picks one keyboard instead of all, moves the drum channel, and chooses where a sound's original pitch sits: C4 for every sound (AudioFinder's way), its own note (a G3 sample sits on G3), or the song's key — the sound as recorded on the tonic of the Chords tab's key, so playing up and down from the tonic hears it in the song's key.

Ask Oscilla: chords_set_pads

## Model categories

Ask Oscilla which sound categories the classifier recognizes.

Ask Oscilla: model_categories

## Morph

Ask Oscilla to make one sound become another through spectral interpolation. Duration and curve shape the transition; a duration of 0 runs as long as the longer source. A stereo pair stays stereo, each channel making the same journey on its own.

Ask Oscilla: get_selection, list_files, morph

## Mosaic

Ask Oscilla to rebuild a target recording from short grains in a folder of sounds. The first run

indexes that folder; later runs reuse its index.

Ask Oscilla: get_selection, list_files, mosaic

## Move scratch out of Recordings

Moves the test files and the machine-named scratch renders older than fourteen days that nothing names out of Recordings into Documents ▸ AudioCortex ▸ Cleared ▸ <date>, with a manifest — a rename on the same volume, never a delete. Subfolders, the person's own takes and DJ sets, and every name on a KEEP*.txt list in Recordings stay. Undo puts the last sweep back. When the songs and projects could not all be read within the given seconds (three minutes), nothing moves and the reply says to run storage first; when one of them cannot be read at all, nothing moves and the reply names it. The list of what a sweep moved is written before the first file moves, and undo puts files back only from that sweep's own folder into Recordings. Settings ▸ General ▸ Storage has the same buttons. The person empties Cleared in the Finder when sure.

Ask Oscilla: recordings_sweep

## Onsets on the bar grid

The detected hits of one to four files in the kick, snare and hat bands, drawn on a beat/bar grid at bpm from offset. Each lane's lean is the median distance of its kick hits from the nearest beat in milliseconds, and the lean against the first file says whether added drums land ahead of or behind it.

Ask Oscilla: onset_grid_image

## Open window

Ask Oscilla to show the room for the work — any room in the sidebar, the editor for a file, or the piano roll. The dock switches in place; a popped-out window comes forward. The Synthesizer Sampler is 'sampler'; the floating Song Map panel is 'songmap'.

Ask Oscilla: get_selection, list_files, open_window

## Pitch

Ask Oscilla for the pitch of a sound: the frequency, the note and how many cents it is off, a confidence, and the sample-root a sampler voice should be given so the file plays in tune. It reads the loudest second of the sound and checks the octave against the sound's own harmonics, so a bell with a strong upper partial still reads as its fundamental. Noise, drums and chords come back with a low confidence and no note.

Ask Oscilla: get_selection, list_files, pitch

## Play

Ask Oscilla to play a file, or play the current selection. This auditions the file without making a new render, and shows it: the file’s row is selected in the list in front of you, opening its folder in the browser if it was not showing, so you can always see what is playing.

Ask Oscilla: get_selection, list_files, play

## Play AudioCortex from a MIDI controller

Settings ▸ Controllers lets the knobs, faders, pads and buttons on a MIDI controller or groovebox play AudioCortex: the Knob Panel's knobs, the synth on the Instruments panel,

each Tracks lane's volume, pan, sends, mute, solo and record-arm, and the transport (play, stop, record, loop, the metronome, the Knob Panel's A/B). To map one, right-click a knob in the app and choose MIDI Learn, then move a control on your device. Or pick a control in Settings ▸ Controllers and press Learn. Turn an endless encoder a few clicks while learning and the app hears that it is endless. A knob on your device takes a control only when it reaches the control's value, so grabbing it never makes a sound jump. The value shows over the window as you turn, and Pick up or Jump can be chosen per mapping. A pad mapped to a button presses it instead of playing a note; unmapped notes still play as before. Fine-resolution controls (14-bit pairs and NRPN) and pitch bend work too. A row of your device's knobs can follow the Knob Panel — whatever knobs Oscilla wires, your knobs turn them — with two buttons for the next and previous page. While you are turning a knob on your device, Oscilla's changes to it wait for your hand; and when a value changes in the app, a device's knob mapped to it hears the new value, so a motorised fader or an encoder ring follows. Switch on Write automation and, while Tracks plays, a knob on a lane's volume, pan or send draws the lane's automation line as you turn — what you pass over is replaced, the strip goes back where it stood when you let go, and Undo takes the pass back. Under MIDI Out and Clock, turn the clock on for a device and it plays in time with Tracks — it starts, stops and follows the song's tempo and position; if its sound comes back late, move its clock earlier. Or choose a device under Follow the clock from and Tracks follows it instead: its Start plays the set, Continue carries on from the position it sends, Stop stops, and the song's tempo follows its clock. A lane can have a hardware instrument behind it (Oscilla attaches one with a MIDI file and the device): the notes play the device while Tracks plays, Record on the lane prints the device's sound onto it, and Hardware Lanes here shows them with a Live switch for the notes. A Knob Panel knob can also turn a knob on the device itself, so one knob in the app can move the app and your hardware together. The MIDI Monitor in the same tab shows what your devices send, and the tempo of a clock they send. Mappings are kept in Documents/AudioCortex/Controllers.

Ask Oscilla: clock_set, controllers_learn, controllers_list, controllers_map, controllers_monitor, controllers_send

## Process

Ask Oscilla to apply edits such as trim, reverse, fades, gain or stretch. Edits run in order and write a new file; the source stays intact. A file too long to hold whole (a thirty-minute bounce) is still cut: put the trim first and only that window is read.

Ask Oscilla: get_selection, list_files, process

## Recordings, songs and what a song keeps

Recordings is the folder where AudioCortex puts what it makes outside a song: a render from the Nodes canvas, a bounce of a loose set in Tracks, a tone, a kept copy of a file dropped from a folder that clears itself. It is Documents ▸ AudioCortex ▸ Recordings, and the folder item at the foot of the dock opens it; the number on that item is how many files have landed there since you opened the app — a count of what this sitting made, not a recording light (nothing is recorded live; to record, press Record in Tracks or the Arranger). The first click explains it; right-click the item for this entry or to show the folder in Finder. A song is the other place, and the one to keep. Save a set as a song (Tracks ▸ ⋯ ▸ Save as Song…, or ask Oscilla) and it becomes a package in Documents ▸ AudioCortex ▸ Projects that carries everything of its own: the lanes as its arrangement, its stems, every bounce as a numbered render (v1, v2 …) on its record, its Nodes patch, knobs and Song Map, its moments (File ▸ Checkpoint…) and the history of every change by you, by an agent or from the phone (File ▸ History…). While the room shows a song, Bounce saves into the song's Renders folder as the next version and

tracks_bounce does the same; while the set is loose, both go to Recordings and a checkpoint goes to Documents ▸ AudioCortex ▸ Checkpoints. Your own sound files are never moved: a song references them where they are until Collect All Files into the Song copies them in. In short: Recordings is the bench, where things land while you work; a song is what you keep, with its own history.

Ask Oscilla: tracks_bounce, project_bounce, project_save, project_collect

## Remove from the Crate

Remove file references from the Crate. The files remain on disk.

Ask Oscilla: bin_remove, get_selection, list_files

## Resonate

Ask Oscilla to excite one sound with another sound's resonances. Choose a striker and body, then set ringing and blend.

Ask Oscilla: get_selection, list_files, resonate

## Reveal in the Finder (tool)

Oscilla can show you where a file or folder is: a Finder window opens with it selected. Nothing is moved or opened. Ask after a render, a bounce or a take to see where it landed.

Ask Oscilla: reveal_in_finder, open_window

## Reverb

Ask Oscilla to convolve a recording with another sound as its room response, or use a synthesized hall. Mix controls the wet fraction.

Ask Oscilla: get_selection, list_files, reverb

## Rooms and shortcuts

Click a dock icon to switch rooms. Command 1–9 chooses the first nine dock items, including the Explorer’s separate window. Control-Tab and Control-Shift-Tab cycle the embedded rooms. Option-click opens a room in its own window. Double-click the rail to change its compact appearance; a short window shrinks the rail's items together so every room stays in reach, and resting on an item names it at once. Window ▸ Workspace chooses which rooms the rail shows (see Workspaces). Use the question mark for all Help, the info button for the current room, or a dock item’s Help menu for that item. The title-bar ? search still searches sounds. If AudioCortex was opened from the Downloads folder or straight from the disk image, it offers to move itself into Applications and open again from there — updates can only be installed once it lives in Applications. View ▸ Hide Oscilla Pane (⌥⌘L) toggles the Oscilla pane. View ▸ Play Through the List (⌥⌘→) toggles play-through in the Library. Help ▸ Our Promise opens Iced Audio's promise page: no AI makes the audio, and your sound stays on your Mac.

## Scratchpad

Oscilla's plan on a scrap of paper for this run of the app: it sets the pad (text), adds a line (append) or reads it back (no arguments). Before a long job Oscilla writes a three-line plan here and reads it again after many tool rounds — as a session grows, the oldest tool results leave the model's context to make room, and the scratchpad does not. It clears when the app quits; agent_note is the memory that lasts across sessions, session_note the journal that

travels with a piece.

Ask Oscilla: scratchpad, agent_note, session_note

## Search

Type words in the title-bar search to filter the current list by name, category and embedded description; clear it to restore the list. Begin with ? to describe a sound (the analyzed sounds in the list are ranked by what they read as). Begin with * to search EVERYWHERE — every analyzed sound on this Mac, not the list in front: “* kick” finds every sound named or classified kick, “*? deep bass” describes a sound across them all; the results come as a list of their own with each sound's category, and the line under the browser says how many of the analyzed sounds matched. For Oscilla the tool is search {query, everywhere}; library_search returns the same rows without touching the Library.

Ask Oscilla: get_selection, list_files, search, browse_folder, library_search

## Search by text

Begin the title-bar search with ? to describe a sound. Results are ranked by text-to-audio similarity. If the sound-search helper is unavailable, use name, category or reference-sound search.

Ask Oscilla: get_selection, list_files, search_by_text

## Search every analyzed sound

Ask Oscilla to search every sound this Mac has analyzed — all of them, across every folder, collection and scan set — not only the list in front of you. By name: every word must appear in the file's path or its sound category. By sound: the words name sound categories (bass, kick, pad, vocal, bell…) and the analyzed sounds are ranked by how much they read as those, the ? search everywhere. It reads the app's own stores and opens no file, so it answers in a moment over hundreds of thousands of sounds; the reply says how many sounds are analyzed and carries each hit's category and confidence. search {query, everywhere:true} shows the same results as a list in the Library. A sound never analyzed is not in the store, except by name: files in the list in front of you are found by name straight away and come first. To search one by sound, browse its folder (folders analyze as they list) or call analyze. The word unanalyzed alone shows the sounds the classifier has not heard yet.

Ask Oscilla: library_search

## Search the index

Search a library's index instead of its disk — instant over a share: a name fragment anywhere on the path, or every file, and on an analyzed index a tempo (half and double counted) or a key. The folder can be the library or any folder inside it. find_files does the same by itself whenever the folder has an index.

Ask Oscilla: index_search, find_files

## Sediment

Ask Oscilla to repeatedly pass a sound through a room response. Later generations emphasize the room's resonances. Works on the first two minutes of a source and says so when it cut.

Ask Oscilla: get_selection, list_files, sediment

## Select file

Ask Oscilla to select a file by part of its name or its row. Selection auditions the file and starts quiet tempo and key analysis.

Ask Oscilla: get_selection, list_files, select_file

## Set comment

Ask Oscilla to write a comment on a file. This updates its annotation, not its audio.

Ask Oscilla: get_selection, list_files, set_comment

## Set rating

Ask Oscilla to rate a file from zero to five. The rating is saved with its annotation.

Ask Oscilla: get_selection, list_files, set_rating

## Settings

Settings (⌘,) has six tabs: General, Audio, Appearance, Phone, Controllers and AI. General: appearance, spectrogram colors, Ask before quitting (on by default: ⌘Q shows “Quit AudioCortex?” first, and its Don’t ask again box turns it off here; an update installing, or the Mac logging out or shutting down, never asks), automatic classification tags, waveform-cache location and DSP output location (Empty Cache removes cached previews), the Output Device (every room plays through it; System Default follows the Mac's own setting; Audio MIDI Setup… opens Apple's own panel for the device's sample rate, clock and aggregate devices, Refresh reads the list again after plugging an interface in; when the device in use is a Bluetooth headset or speaker, an amber note under the picker says so: Bluetooth delays sound by about a fifth to a third of a second, playback is fine, but beats you play or tap and the lit step on the drum grid feel late — for playing, recording and timing use wired headphones, the Mac's speakers or an audio interface; the first time sound goes out over Bluetooth in a run, an alert says the same — “You're listening over Bluetooth” — with an Audio Settings… button and a Don't warn me about Bluetooth again box; the switch Warn when sound goes out over Bluetooth under the picker turns the alert back on), the Input Device (what Record in Tracks, the Synthesizer Sampler and Live Listening listen to, unless their own pickers name another device; System Default follows the Mac; its own Audio MIDI Setup… and Refresh), MIDI In (play from a MIDI keyboard on or off, which keyboard — All Connected by default — and the channel the drum machine answers on, 10), and the audio buffer size: smaller answers quicker when you play keys or pads but works the Mac harder, larger is steadier, go up one step if you hear clicks; it takes effect when engines restart, and a Bluetooth output adds far more delay than any buffer. A sound with more than two channels auditions as a stereo fold-down (odd channels left, even right). Appearance: the color scheme — Glacier (the default), Aurora, Ember and Mono, one color for sound and one for what is moving, applied to every open room at once and remembered — and Text size: Default, Larger or Largest, one factor on every room's labels and numbers, applied at once; a larger size makes a dense room taller and wider, so give its window more room if a row wraps; the Library's lists and the Mac's own dialogs keep the system size. AI: the first line says which of Oscilla's five options is in use right now and why; then a choice: Automatic (the first option that is set up) or a button on each option to pick it yourself — a pick that is not set up falls back to Automatic; then the five options in the order Automatic tries them, each marked set, not set or in use: 1 Claude (Anthropic key or sign-in), 2 Gemini (key and model), 3 your own model server (address, model name, optional key), 4 Oscilla Local (Iced Audio's downloadable model with its own version: Download, Update, Check Again, Remove), 5 Apple's on-device model (available or not). Keys are stored

in Keychain. Below them, the optional sound-search helper address, which enables description search and related scoring; the rest of the app does not depend on it. A sound's original pitch sits on C4 for every sound (AudioFinder's way); Settings ▸ Audio ▸ MIDI In can put it on the sound's own note instead, so a G3 sample sits on G3. The Phone tab turns on the phone remote and shows the code to scan. At the top of the AI tab, “Who answers your notes?” picks who answers the notes you leave on the Song Map, the Chords tab, a Tracks lane, the phone or a sung melody: Oscilla (in this app) answers right away; another assistant (Claude Desktop, Codex…) sees them the next time it does something. Controllers: MIDI learn, the mappings from your controller's knobs, faders and buttons to controls in the app, and a monitor of what your devices send. The Audio tab's input meter asks to use the microphone the first time, and says so while the question is open.

## Slice at beats

Ask Oscilla to slice a file at detected beats. Each beat becomes a separate WAV. A half-time grid is doubled when that lands in the dance range; `bpm` names the grid you mean. The reply names the folder, the count and a manifest file listing every slice, with only the first twelve paths inline. Pass `bpm` when the tempo is known; the slices follow it instead of the detector's.

Ask Oscilla: get_selection, list_files, slice_at_beats

## Slice file

Ask Oscilla to chop a recording at transients. Threshold controls detection and the release gap separates hits. Each slice is a new WAV. The threshold is relative to the file's own peak.

Ask Oscilla: get_selection, list_files, slice_file

## Spectral cross

Ask Oscilla to impose one sound's spectral shape and dynamics on another. The carrier repeats to cover the modulator.

Ask Oscilla: get_selection, list_files, spectral_cross

## Spectrogram on an absolute scale

A file's spectrogram as a picture: color is dBFS from –100 to 0, the same for every picture (no per-image auto-gain), on a log frequency axis from 20 Hz to 20 kHz so the sub and bass rows have room, with a labeled time axis. compare stacks a second file below on the same scale. Beside the picture, per section: each band's share of the energy — sub 20–80 Hz, bass 80–150, low-mid 150–500, mid 500–2k, presence 2k–6k, air 6k–20k — with crest and LUFS, and a warning when the sub holds the mix or the crest drops under 8 dB. It reads up to an hour of a file; a section that lies outside what was read is left out, and a warning names it.

Ask Oscilla: spectrogram_image

## Spectrum

Ask Oscilla to inspect an audio file's frequency balance. Bands show energy shares; peaks show frequency, nearest note and cents. Give from and to in seconds to measure only that passage. The band shares sum to 100; each bin is counted once.

Ask Oscilla: get_selection, list_files, spectrum

## Spectrum live

Ask Oscilla to measure the named live synth voices and optional drums. It records a short sample to analyze.

Ask Oscilla: get_selection, list_files, spectrum_live

## Speed

Ask Oscilla to change a file's speed. Pitch follows speed, like varispeed playback.

Ask Oscilla: get_selection, list_files, speed

## Status

Ask Oscilla what Library is showing before changing it. While a Deep Scan runs, status carries scan: the folder, how many sounds it has found, how many folders are still to enter.

Ask Oscilla: status

## Stereo field over time

Left/right correlation over time (1 is mono, 0 unrelated, –1 canceling), the width (side against mid), the correlation of the low end alone (under 150 Hz), and a goniometer per section. Warnings when the correlation stays negative for a second or more, or the low end is not mono-safe.

Ask Oscilla: stereo_field_image

## Stop

Stop file preview, the Tracks room and the Nodes canvas. If synths or decks are still sounding, ask Oscilla to stop everything.

Ask Oscilla: stop, stop_all

## Tags on sounds

Your own tags on sounds, as AudioFinder had them and more. The tags live in groups you name, and a tag can have spaces. Put one on a sound from the Library's right-click menu (Tags lists them, ticked for the sounds that carry them; New Tag makes one for the chosen sounds), or ask Oscilla to tag files one at a time or in bulk. The Tags heading in the Library's sidebar lists every tag with how many sounds carry it; choose one and the Library lists those sounds. In the search field, #tag or tag:"two words" finds a whole tag, never a part of one. Window ▸ Tags… is the vocabulary: add, rename or delete tags and groups; renaming or deleting a tag asks once whether to change the files that carry it or only the list. Prune Missing Files drops files gone from disk (a drive that is only unplugged keeps its files). Import from AudioFinder brings AudioFinder's tag list and the tags on its files into a group of their own, once, so nothing is tagged twice; AudioFinder is not changed. Tags stay in the app's own list; switch on Finder tags and from then on they also go on the files as Finder tags.

Ask Oscilla: tags

## Texture fingerprint (what makes a recording sound the way it does)

Ask Oscilla what makes a recording sound the way it does. It measures the things a sound designer listens for — how wide it is, how dense, where the energy sits and how dark or bright that makes it, which notes it holds, how slowly it breathes — and answers in a plain sentence with the numbers behind it. Give it two recordings and it tells you how the second differs from

the first: a mix against a reference, or a cloud you made against the track you were aiming at. It only listens; nothing is changed.

Ask Oscilla: texture_fingerprint, lush_stretch, detect_bpm, detect_key

## The audio editor

Double-click a sound (or ask Oscilla) to open it in the editor; the volume knob, the time, then Loop, Stop and Play sit at the bottom right, and Space plays and pauses. Loop goes round the selection, or the whole sound when nothing is selected, and takes effect the moment it is pressed. Drag across the waveform to select; hold ⇧ to extend the selection, Delete removes it and closes the gap, Undo brings it back. The orange bars at the start and end are the trim handles — drag them in to cut the beginning or the end, the parts cut off shade over — and the small squares on the top edge are the fades: drag to lengthen a fade in or out. Fade In and Fade Out set those squares over the selection, or up to the playhead. None of this changes the file: Play hears the trims and fades at once, and keeps playing while you drag them; Save (or Process ▸ Apply Trim & Fades) writes them in. Double-click a handle to put it back. Zooming — the zoom bar, the ↑ and ↓ keys, a pinch — keeps the selection in the middle of the view when there is one. Markers are bookmarks: set one at the playhead (or let Detect Beats or a slice set them) and it is kept beside the sound, in Documents ▸ AudioCortex ▸ Marks, the moment it is made — the file itself is never touched — and it is there again the next time the sound is opened; Save also writes the markers into the file as cue points for other apps. To take a piece of a long recording, select it and drag the selection out of the window: it lands wherever it is dropped — a folder, a DAW's track — as a rendered WAV named after the sound and the times ("bed chop 1m23.4s-1m40.0s.wav"), it carries where it came from as the file's comment, and the original keeps a shaded band where the chop was cut. Opening a chop in the editor shows From ‹the sound› at the top: a click goes back to the original with the cut selected.

Ask Oscilla: open_window

## The Metadata window

Ask Oscilla to show the Metadata window — AudioFinder's metadata viewer, returned: the eleven embedded fields for the Library's selected sound in a floating panel, with Guess (tempo, beats, time signature and key from the sound), Revert and Save into the file. It follows the Library's selection while open. Under the fields sits the session tempo an Option-drag from the Library renders a loop at.

Ask Oscilla: metadata_window

## The sound index

The sound index is a catalog of sound fingerprints (two files — the fingerprint matrix and its list of paths — in the Models folder under Application Support) that the search-by-ear tools read — find_similar over a corpus and search_by_text. It is not part of the download: it is built from a sound collection with Iced Audio's own tools, for the desk, and a tool that needs it says so in one line rather than failing. Everything a person uses works without it — browsing, playback, analysis and classification (the sound model itself is in the app), the Library's name search, the ? describe-a-sound search, the * search across every analyzed sound, and every engine and room. “Not installed” in Knowledge ▸ Model Details means only that; there is nothing to fetch.

Ask Oscilla: find_similar, search_by_text

## Tonal balance against references

The file's average spectrum drawn over the average spectra of one to four reference files,

every file brought to the same loudness first, so the curves compare fairly. Per band — sub, bass, low-mid, mid, presence, air — the file's difference from the references in dB, and a warning when a band sits more than 4 dB above them. sections gives the same differences per stretch.

Ask Oscilla: tonal_balance_compare

## Veil

Ask Oscilla to separate audible features from a masked residual. Heard keeps the exposed material; hidden keeps what was masked.

Ask Oscilla: get_selection, list_files, veil

## Vocal remove

Ask Oscilla to separate centered vocal content from a stereo mix. Centered instruments can be affected too; mono and off-center vocals do not separate cleanly.

Ask Oscilla: get_selection, list_files, vocal_remove

## Waveform image

Ask Oscilla to inspect a file's waveform. The image exposes transients, silence and overall structure.

Ask Oscilla: get_selection, list_files, waveform_image

## Wavesets

Ask Oscilla to repeat, omit, reverse or shuffle wavesets between zero crossings. Repetition lengthens the sound; omission shortens it.

Ask Oscilla: get_selection, list_files, wavesets

## What is analyzed, folder by folder

Which folders the classifier has heard and which it has not: under a folder, every subfolder down to a chosen depth with its sounds, how many are analyzed, how many are not, and the share — the biggest backlog first — plus the totals for the whole. The walk has a time budget (a minute) and says when it stopped short. For the list in front of you the Library's summary strip shows the same two counts, and its unanalyzed count is a button: press it and the list shows only the sounds not analyzed yet (typing unanalyzed in the search field does the same; press again for everything).

Ask Oscilla: analysis_overview

## What Recordings holds

How much of Recordings a song, project, set, checkpoint, kit or crate actually names, how much is the person's own takes and DJ sets, how much is recent scratch, and how much is test files and machine-named scratch renders older than fourteen days that a sweep would move — with the size of the Cleared folder and the sweeps in it. The app reads every song and project to find what is named: seconds the first time, less after, because what it read is remembered. A huge Music folder stops the reading at the given seconds (forty-five), and the report then says so and offers nothing to move until a later call finishes. A song or project file that cannot be read at all holds the sweep back the same way, and the report names it.

Ask Oscilla: storage

## Widen

Ask Oscilla to widen a mono render. Complementary frequency-band levels spread it in stereo while keeping the low foundation centered. The width you ask for is the width the result will measure; when the bass kept below keep_low_hz holds the correlation up, the reply says what this file can reach and that lowering keep_low_hz goes further.

Ask Oscilla: get_selection, list_files, widen

## Workspace

Ask Oscilla which rooms are on the rail, or to change the workspace: Everything, Browsing, Making, Performing or Custom, or a Custom list of rooms. The same as Window ▸ Workspace. A hidden room is still shown by open_window and joins the rail for the sitting.

Ask Oscilla: workspace, open_window

## Workspaces: which rooms the rail shows

A workspace is the set of rooms on the main window's rail, and AudioCortex opens in the one you chose. Window ▸ Workspace lists four presets and Custom: Everything (every room), Browsing (Library, MIDI, Sounds, Explorer), Making (Library, Sounds, Instruments, Arranger, Tracks, Nodes) and Performing (Library, DJ, Instruments, Spiral). Settings ▸ Appearance ▸ Workspace has the same picker and a button for every room; turning a room on or off there makes the workspace Custom, which keeps your own set. The Library is always on the rail. A hidden room is still a room: Oscilla's open_window shows it and Window ▸ Workspace reaches it, while Control-Tab steps through the rooms on the rail. Once shown, it stays on the rail until you quit. Ask Oscilla to change it with the workspace tool, which also reports what the rail shows now.

Ask Oscilla: workspace, open_window

## Write embedded metadata

Ask Oscilla to write a sound's metadata into the file the way AudioFinder does: the fields given are merged over what is there, an empty value removes a field, and the file's modification date is kept so nothing re-sorts. With guess true the tempo, beats, time signature and key are read from the sound first and written — the Library's Guess button does the same. Only WAV and AIFF files carry it.

Ask Oscilla: metadata_write

# MIDI Library

midi_scan indexes, midi_search finds, midi_read inspects, midi_perform plays.

## MIDI Library

Scan folders of MIDI files, then filter by key, tempo, role, feel or density. Read a file before assigning sounds. Performance through the app uses synth voices and drums on channel ten. It honors the first tempo rather than following a changing tempo map. Click a file to hear it and to read its chords. The headline tempo, feel and key carry their counts. The summary strip shows the library's count with a bar split between tuned (the sound color) and percussion (the motion color), a small tempo histogram with the median marked, the feel and the top keys; the filters are hex-cut chips; each row carries a dot in its color, hex pips for density and the parts as colored chips; the chosen file's chords are hexes, with a Play button beside them: a plain hexagon with an amber triangle at rest, lit amber with Stop while the file plays, as everywhere else in the app. The space bar plays the selected file and stops it, and the up and down arrows move to the next or previous file and play it, as a click does, wherever the focus is in the room. In the search field the arrows still step through the list, while the space bar types a space; Oscilla's prompt keeps both keys. A file that has moved or been deleted since it was indexed is skipped. Click a column title to sort the list by it, name, tempo, meter, key, feel, density or parts; click it again to reverse the order. The preview plays through the app's own engine, on the output chosen in Settings like every room; if it cannot sound, the reason is written beside the Play button instead of silence. Clicking a file takes the keys back from the search field or Oscilla's prompt, so Space and the arrows work on the list.

Ask Oscilla: midi_perform, midi_read, midi_scan, midi_search

## MIDI perform

In MIDI, choose a file and ask Oscilla to perform it with the app's instruments. Channel ten plays drums; other channels receive synth voices. Use offline to render without playback, and stems to save each instrument, fallback pool and drums as aligned WAVs beside the mix in Recordings. Only the first tempo is used; bpm overrides it.

Ask Oscilla: midi_perform

## MIDI read

Ask Oscilla to inspect a MIDI file before choosing instruments. Notes and timing are read from the file; roles and key are inferred.

Ask Oscilla: midi_read

## MIDI scan

Scan a folder in MIDI to index its musical contents. A rescan rereads changed files.

Ask Oscilla: midi_scan

## MIDI search

Filter the indexed MIDI library by key, tempo, role, feel, density or filename. Scan a folder first. Files on a volume that is not mounted right now stay in the index but are listed apart as offline, with the volume’s name, so a plan is never built on files that cannot be opened. The limit counts playable matches; stale entries never crowd them out.

Ask Oscilla: midi_search

# Musical Judge

## Aesthetic

Ask Oscilla to measure a bounce's perceptual qualities, such as warmth or harshness. This needs the sound-search helper.

Ask Oscilla: aesthetic

## Judge

Ask Oscilla to check a render for clicks, level problems and tonal content before presenting it.

Ask Oscilla: judge

## Judge musical

Ask Oscilla to check a bounce against your brief. It reports mix, rhythm, arrangement and harmony issues with locations in time. Give from and to in seconds to measure only that passage. Give the bpm you know the piece is at and the grid test uses it; otherwise the judge checks its own tempo guess against the ears' estimate and tests the grid the onsets fit better, and says which it used. A long or high-rate file is brought down to size for the judgment instead of being refused. The reply's measured {seconds, fileSeconds, cut} says how much of the file was read (up to 25 minutes); compare renders only when cut is false for both.

Ask Oscilla: judge_musical

## Musical Judge

Developer builds only: not in the Help menu of the app you downloaded. This developer window checks an audio file against a brief. Drop the file, enter the brief and press Judge. Read the mix, rhythm, arrangement and harmony results, localized issues and revision advice. Measurements guide revision; they do not choose your taste.

Ask Oscilla: judge_musical

## Sounds like

Ask Oscilla to compare a recording with a few written descriptions. Include the intended sound and an unwanted alternative.

Ask Oscilla: sounds_like

# Nodes

nodes_get reads the canvas; nodes_set replaces it; nodes_render runs it.

## A render, rebuilt from its recipe

Every render from the Nodes patch keeps its recipe beside it — the patch as it was when the render began. Rebuilding renders that recipe again, with the live patch put straight back, and measures how closely the new render matches the old: their difference against the original, where –60 dB and below is the same sound.

Ask Oscilla: render_rebuild, nodes_render

## AudioUnit validation

Before AudioCortex hosts a third-party AudioUnit for the first time, it runs Apple's validation (auval) on it, out of process, and keeps the verdict until the unit's version changes — the way Logic and Live do it, so the wait is one per unit, not one per render. A unit that fails is kept out and named with auval's last lines wherever it would have played: a Nodes card, a Tracks lane, a bounce. Ask Oscilla for au_validation to see every verdict, to run the check again on one unit (recheck), or to let a failed one in anyway or keep a passing one out (allow). Apple's own units are not checked. A check that runs out of time is not a failure: the unit is hosted under the crash guard and the verdict says so.

Ask Oscilla: au_validation

## Every kind of Nodes card, described

What an agent reads before building a patch: every card the app has, with its key (what nodes_set takes), name, family, the one-line blurb and the full story, its inputs — port 0 the main, port 1 the partner with its label and whether it is optional, a mod port where an OSC may go — whether it has an output, every parameter with its default, range and help, and the choices where a card has a picker. kind narrows it to one card.

Ask Oscilla: nodes_kinds, nodes_set, nodes_get

## Generate a part

Ask Oscilla to generate a part as a file — the parts that used to need a synth script outside the app. Six recipes: a supersaw lead (four band-limited saws with a spread, glide between the notes and a vibrato that comes in after a moment, playing a phrase of eighths), a sub (a sine an octave under the roots you name, one root a bar, ducked a few dB on every beat where the kick sits), a rolling bass (eighths on the roots where the accented steps open the filter), a kit groove (the stock kit or a folder of one-shots on a backbeat, half-time, four-on-the-floor or breakbeat pattern, with swing, slightly human hits and a tom fill every few bars), sparkles (bell notes scattered over the key's pentatonic) and a riser (partials an octave apart gliding upward together so the rise never seems to end, with noise opening under it). Every recipe renders through the app's own synth voice and drum machine, offline, into Recordings; the same seed gives the same file, so a groove you liked comes back. Nothing here is a model making sound — the recipe is the musician's numbers, rendered. A run is at most 64 bars; asking for more gives 64 and the reply says so. A kit folder's one-shots are read by slot name in either spelling (hatClosed.wav or hat-closed.wav, wav/aif/aiff/flac); the reply's label says which slots came from the folder and which are missing — a missing slot plays the engine's own voice.

Ask Oscilla: generate, synth_render, drum_pattern, nodes_set

## Hear a proposal

Render a proposal and play it, with the live patch put straight back — nothing on the canvas changes.

Ask Oscilla: proposal_audition

## Mix Narrowcaster: the small mono speaker, the translation check

The translation check: the small single-driver mono box mix engineers keep beside the main monitors and switch to, to hear whether a mix survives — the two sides summed to mono, nothing under 120 Hz or over 8 kHz, a mid-forward tilt, a little box resonance and a little grit. The constraints make a mix's problems obvious: a vocal that does not carry, a kick lost under the bass, a wide pad that cancels in mono. mix_narrowcaster writes a file through it beside the source and says what the narrowcaster reveals: what the fold to mono loses in dB, the share of the energy in 1–4 kHz before and after (where a voice carries), the band shares, LUFS, warnings. The same narrowcaster is a Nodes card — with a live frequency scope while the set plays, and the spectrum before and after it from the last render — and an AudioUnit, Iced Audio: Mix Narrowcaster, for a Tracks insert or the master, so a whole set can be heard through it.

Ask Oscilla: mix_narrowcaster, spectrogram_image, play

## Multisamples and SoundFonts

Ask Oscilla what a multisample holds before a Multisample card plays it. A SoundFont (.sf2) lists its presets in file order — the card's preset number is the index here — and the chosen preset's zones: each sample's name, key range, root and loop points. A folder of notes lists the zones found: one per root, the root read from each file's name (C3, F#4, 60) or, failing that, from the sound. The Multisample card in Nodes then plays either as an instrument: a note takes the zone whose key range holds it, else the nearest root, through the sampler with that zone's sample, root and loop. Alan Wilder's Emax banks came to the library as SF2 beside EB2; the SF2 plays. Emax II banks (.EB2) are a proprietary layout and are not readable yet — convert them to SF2 with a bank tool.

Ask Oscilla: multisample_presets, nodes_set

## Nodes

Build a processing graph from source, effect and output cards. Add Node lists every card with a line on what it does (a search box narrows it; the list scrolls); the (i) on a card tells the whole story and what each knob means; a card's ✓ means its render is current (hover says so), a second input port is named beside it (room, becomes, carrier, body), and Clear empties the whole canvas while a card's × removes that one. Connect the wires, then render to run the graph; a wire that would make a loop is refused. Unchanged nodes reuse cached results — across renders and across Oscilla's rebuilds of the patch: a card whose knobs, choice, file and inputs are unchanged keeps its last file — unless a file it reads has changed on disk, which renders it afresh — so a level change on one card renders only that card and what follows. Synth and Sampler cards render offline, in seconds. A card's render goes into the project's Renders folder (Documents/AudioCortex/Nodes Renders without a project), never beside your source sounds. A card reads up to an hour of a long sound (the spectral cards, such as Reverb and Freeze, up to fifteen minutes); when a sound runs longer, the card uses its start and the render says which sounds were cut, so a piece is never shortened without a word. The rendered file plays and appears in Tracks; Play lights while it plays. An oscillator card modulates a supported effect input; it makes no audio by itself. A Plug-in card runs any

AudioUnit installed on this Mac, with its own window for settings, saved with the patch. Spiral and Spectrum cards are windows on the signal where they sit — a picture after each render — and pass the sound through. The Mix Narrowcaster card is the small mono speaker mix engineers keep for translation — mono, 120 Hz–8 kHz, mid-forward — with a live frequency scope while the set plays. The Sampler card plays the sound on its input as an instrument through the synth's sample oscillator: type notes on the grid, a chord or a MIDI file; the root is read from the sound unless given. The Paula card is the Amiga's sound chip: the sound on its input is made 8-bit and played by ProTracker period — every note lands on the chip's integer periods, from one octave below the root to just under two above — at a 6-bit volume, with the hardware loop (a start and a length in samples), no interpolation (the aliasing is the sound), the power-LED filter and the chip's 28 kHz output when asked; the same notes text and notes port as the Sampler. The Ear card is the ear meter as a card: the part on its main port against the rest on its second — how much of the part is heard, what masks it and where — with the two mixed as its output (the part alone with nothing on rest). A faint grid lies under the cards. A Note card holds words — what the patch is for, where to start — with no ports and no sound, saved with the patch. Every card wears a color bar for what it does — the Colors menu picks a palette (Hive, by function, warm, cool, plain) and right-click a card's name gives that card its own color, saved with the patch. The dice makes a random patch from a sound in the Library. Drop sounds from the Library onto the Nodes item in the sidebar and each becomes a Source card. An empty canvas says what a patch is, offers a button that places a Source and an Output to start from, and two things to ask Oscilla. Several wires into one port are heard together, and every card's output is held under full scale. Psy-Loop is a tape loop: the input is recorded into a loop and played forward, backward, slow or fast, layering over itself; an OSC or a file on its wobble port bends the speed. Structure cards sit first in Add Node: Loop repeats a sound to a length with a crossfaded seam; Place starts a branch later and at a level — with wires into one port summed, that is how branches are lined up into a whole. Arpeggio turns any sound into a phrase: each step replays it at a note of a chord, in a pattern on a tempo grid, sampler-pitched — menus for the pattern and the chord, knobs for tempo, rate, octaves, gate, swing, accent, humanize and transpose. Our instruments are cards too: Synth is a source that plays one note of an analog preset (or, from Oscilla, an FM preset or a wavetable) through the live engine — wire it into Arpeggio for a phrase, into the effects for color. An oscillator's smooth (ms) is how fast its modulation moves — a square gate with it does not click. A card that is given settings that leave nothing says which card and why. Dynamics and Tone: the Envelope card shapes level three ways by its mode — shape (an ADSR on each trigger: an OSC on the trigger port is a trance gate with real edges, a sound there opens the gate with its hits, nothing uses the signal's own onsets), follow (the trigger port's level ducks the main signal — a kick pumps a pad), transient (gain on the hits and on the body between them). The Equalizer is five bands: a low shelf, three parametric peaks with frequency, Q and gain, a high shelf; a band at 0 dB is untouched. The Tone card is a test signal — a sine, a sweep, pink or white noise at a level for some seconds — so any card or plug-in can be measured with the listening tools (the spectrum, the chain trace, the tonal balance). Ember EQ is the character EQ — a steep high-pass, passive-tube style lows (boost and attenuation on separate knobs, so both together lift the lows and scoop the low-mids) and highs, two broad inductor style mid bells, and a drive stage — the same unit that goes on a Tracks lane or the master. Tilt is one knob, dark to bright, pivoting at a frequency you choose. The Limiter finishes a piece: it brings the file to a loudness target in LUFS (–14 for streaming, –23 for broadcast) and holds every peak under a ceiling with a look-ahead, so nothing passes and nothing clicks; the card says what it reached and how deep it cut. Tidy — columns by signal depth, rows ordered so wires run straight — lays the cards out by their wiring — sources left, each card one column right of what feeds it — when a patch has become a heap; a card is always kept with its top on the canvas, so a tall card asked for near the edge slides down with the rest of the patch. Presets, in the

toolbar, holds thirty-three ready chains in six groups (Gates & Rhythm, Clouds & Stretch, Dub & Space, Voices, Mastering, Generative): pick one, drop a sound on its Source card, press Render; the preset's one knob — gate Hz, stretch factor, delay time, LUFS, pitch cents — sits under the toolbar. The Delay card's time can be a note value at the canvas bpm (the field in the toolbar) — 1/8, dotted, triplet, bars — and it ping-pongs, holds its repeats forever, damps them like tape, and takes an OSC on its mod port for warble. Granular is the cloud: grains of a chosen size at a density, read from a position in the sound you set, spray, freeze or scrub with an OSC, pitched and spread; the same seed renders the same cloud. Sing makes a spoken voice sing: every voiced moment is moved to the nearest note of a key and scale — or held on one note — with a retune speed from the hard robotic snap to a singer easing in, vibrato once a note is held, and a sustain that holds the vowels; the voice keeps its own formants. Color: Chorus (detuned copies swimming), Flanger (a swept short delay fed back), Phaser (sweeping allpass notches) — each takes an OSC on its mod port in place of its own sweep — and Lofi, the tape chain: wow, flutter, saturation, bit depth, sample-rate hold, hiss, tone. A plug-in card can be set from Oscilla too: parameters by name, a factory preset, an .aupreset file — so a master through installed plug-ins can be asked for in words. Shimmer is the delay with a pitch shifter in its feedback — every repeat climbs by semitones (+12 the classic, +7 stacks fifths). The canvas wears the Hive look: a faint hex lattice, hex ports (an input's edge slate, an output filled with its wire's color, a mod port amber), audio wires in cyan and modulation wires in amber — the key in the corner says which is which. Tidy lays the cards out in columns by signal flow, with room between; a patch Oscilla builds without places, or with cards too close, is laid out the same way.

Ask Oscilla: nodes_get, nodes_render, nodes_set

## Nodes get

Ask Oscilla to inspect the patch currently on the Nodes canvas before editing it.

Ask Oscilla: nodes_get

## Nodes plug-in

Hosts a plug-in card's AudioUnit and lists its parameters — address, identifier, name, value, min, max, unit — with the factory presets and the current one; node is the card's id from nodes_get (n2). Names any of the card's params that matched nothing. The values are pushed again until the unit reads them back; what a unit still does not take after being initialized and eight pushes is listed under notTaken with what was wanted and what it reads, so a card is never called configured while the unit sits at its defaults. Set values with nodes_set's plugin.params (by name, identifier or address) and a preset with plugin.preset; an .aupreset file goes in plugin.aupreset.

Ask Oscilla: nodes_plugin, nodes_get, nodes_set

## Nodes preset

Fills the Nodes canvas from one preset — name is its id or title (a unique fragment works) — and shows it; file is the sound for its Source card(s); knob sets the preset's exposed value (gate Hz, stretch factor, delay seconds, LUFS, pitch cents, bpm, speed). Then nodes_render. The same menu is in the room's toolbar as Presets, and the knob appears under the toolbar after a load.

Ask Oscilla: nodes_preset, nodes_presets, nodes_render, nodes_set

## Nodes presets

Lists the Nodes room's Presets menu: thirty-three ready chains in six groups — Gates & Rhythm, Clouds & Stretch, Dub & Space, Voices, Mastering, Generative — each with its one exposed knob and whether it needs a sound on its Source card (the Generative and Heartbeat presets start from a Synth card and render as they are).

Ask Oscilla: nodes_presets, nodes_preset

## Nodes render

Press Render in Nodes: the result plays and appears in Tracks. Asked through Oscilla's nodes_render tool, the render lands in Tracks without playing; play_layers plays it. A render in flight can be stopped — the room's Stop, stop_all, nodes_render with cancel, or a new patch — and gives up after the card it is on. A render goes into Tracks only when Tracks is empty or already holds a Nodes render; with a song's lanes in Tracks it plays on the patch's own player and the lanes are left alone (to_tracks forces either way). A knob turned on the Knob Panel or the phone re-renders the same way. Every render keeps its recipe beside it (the file with .recipe.json): the patch, the tempo and when — enough to rebuild or explain it. A render that takes longer than the wait (twenty seconds unless asked otherwise) goes on as a job: the reply is then a job id, and job_status carries the render's own reply when it is done, so a long graph never times out a client; wait 0 or background true makes it a job at once, and cancel stops a job as it stops a render. If Oscilla's last patch was refused, a render says so instead of quietly rendering the patch from before it.

Ask Oscilla: nodes_get, nodes_render, play_layers, stop_all, job_status

## Nodes set

Ask Oscilla to build a patch in Nodes. This replaces the current graph and shows it; render separately to hear it. A patch holds up to 300 cards and 1,200 wires. Each card's knobs are its params (the tool lists them by kind; a morph's seconds of 0 means as long as the longer source). A param a card does not have is refused, with its real ones named. Several wires into one port are summed equal-power — an implicit mix. Every card's output is held under 0 dBFS; the render reply says which cards were held and by how much. Loop (seconds, crossfade) repeats a sound to a length; Place (start, db) delays a branch and sets its level. Arpeggio (choice = pattern, chord by name, bpm, rate, octaves, gate, swing, accent, humanize, transpose, seconds) plays a sound as a chord phrase. synth is a source card: choice names the sound (preset:Nova Pad, fm:E.PIANO 1, library:Choir), params note, seconds, velocity. osc takes smooth (ms, 5 by default) so a square gate does not click; fadetrim's trimEnd is seconds cut from the end, not a time, and the render says so when the cuts leave nothing. envelope (mode shape | follow | transient; attack, decay, sustain, release, hold, depth, smooth, transientDb, bodyDb; port 1 the trigger) and equalizer (lowHz/lowDb, b1–b3 Hz/Q/Db, highHz/highDb) are cards too. tilt (tilt dB, pivot Hz) and limiter (lufs, ceiling dBFS, release ms) finish the tone family. granular (size, density, position, spray, pitch, pitchSpray, texture, freeze, stereo, seconds, mix, seed, depth) is a card too. chorus, flanger, phaser and lofi are cards too (rate, depth, feedback, mix, stereo, voices, delay, stages, center, range; wow, flutter, saturation, bits, rate, noise, tone). A plug-in card takes plugin {kind or name, params by name/identifier/address, preset by name or number, aupreset path, state}; nodes_plugin lists what the unit has. shimmer (time or a note value, feedback, mix, semitones, shiftMix, damp, depth, tail) is a card too. The reply's ids map your spec ids to the canvas ids (n0, n1 …) that nodes_get and nodes_plugin use. A plug-in kind may be written as numbers or as four-character codes (au:aufx:lpas:appl). Iced Audio's signature moves are cards too (the Signature group): leveler (follow, amount, target, dip, ceiling; optional port 1 takes an OSC or Shape as the ride), swirl (choice a note value or seconds; time, feedback, darken, rotate, sweepLow,

sweepHigh, sweepPeriod, resonance, mix, tail), doublehelix (two inputs; bpm, bars, depth, dark, seconds, phase), smear (at, window, stretch, bands, spin, seconds, fadeIn, fadeOut), tapedive (choice snap or glide; start, seconds, octaves, hold, glide) and reverseswell (at, window, stretch, seconds, hall, level, fade; optional port 1 an impulse response). The reverb card takes predelay (ms) and damp as well as mix and seconds. duck (two inputs: main and the key on port 1; amount, threshold, attack, release, hold), transplant (two inputs: main and the voice on port 1; amount, cap, smooth, floor) and declick (sensitivity, width, amount) are cards too. The Utility group: split (choice low, mid or high; lowHz, highHz — three of them sum back to the sound), midside (choice both, mid or side; width, midDb, sideDb), match (two inputs, the reference on port 1; offset), join (two inputs, the next on port 1; at, crossfade) and sustain (seconds, crossfade, trim). Also speechstretch (rate, window, search — WSOLA, for a voice), formant (factor, smooth, cap) and whisper (highpass, presence, air, steep). The compressor takes ratio, attack, release, knee and a fading amount (amountIn, amountInOver, amountOut, amountOutOver); fadetrim takes auto (1 trims to the first sound); the equalizer's Q reaches 40 for a notch. Modulation follow-ups: an OSC with beats above 0 cycles once every that many beats at the canvas tempo (hz ignored); a Shape with beats 1 reads its points' times as beats, and a sound wired to its port 0 restarts the curve at every onset; a follower card (attack, release, depth, smooth, floor) turns a sound's own envelope into a modulation for any mod or partner port. warp (choice beats, complex or texture; bpm 0 = detected, offset, maxStretch) is a card too: the sound warped to a grid. multisample (no input; file a .sf2 SoundFont or a folder of notes; preset; the Sampler's parameters and notes text) plays a SoundFont preset or a folder of notes as an instrument, each note from its own zone. paula (port 0 the sample, port 1 a note stream; the Sampler's notes text; params root, note, velocity, seconds, volume 0–64, finetune −8–7, period 0 = from the note or 113–856, loop 0 one shot / 1 whole / 2 loopStart+loopLength in samples, led, lofi, bpm, rate, gate, transpose) is the Amiga's Paula chip as a card: 8-bit, the pitch quantized to integer periods, no interpolation; a note past the chip's range stops at the limit and the readout says so. The Rack group: ringmod (the sound times port 1 or an internal sine at hz; mix), fold (fold, symmetry, stages, mix — a sine wavefolder), distortion (choice overdrive | distortion | fuzz | tape | rectify | bitcrush; drive 0–40 dB, tone Hz, bias for even harmonics, mix for parallel grit, autogain 1 holds the level, output dB, keep Hz leaves the lows clean below it — 120 for a bass, lowcut, oversample 1|2|4, bits and downsample for bitcrush; port 1 optional: a modulator moves the drive ±modDb), freqshift (hz −2000…2000 — every partial moved by the same Hz, not a pitch shift; mix, feedback for the endless riser), comb (hz or note, feedback −0.99…0.99 with negative for odd harmonics, damp, mix, stereo cents, depth — a modulator on port 1 sweeps the tuning; a click in is a plucked string), filter (choice lowpass, highpass, bandpass or notch; cutoff, resonance, poles 2 or 4, drive dB, octaves — a modulator on port 1 sweeps the cutoff), phaserot (stages 4–200 allpasses, center, spread, mix, octaves — a modulator on port 1 slides the center), spring (tension, decay, mix, drip — a spring reverb), gate (threshold, attack, hold, release, range; port 1 an optional key), pitchramp (choice tape or keep; from, to semitones, start, seconds, curve — a glide, a tape stop), tokey (choice the scale; key, strength, speed — every pitched moment moved to the nearest note of the key), harmonize (choice the scale; interval1–3 as scale degrees when diatonic is 1, else semitones; key, detune, spread, mix), vocoder (port 0 the carrier, port 1 the voice; bands, attack, release, lowHz, highHz, unvoiced, mix), spectral (choice freeze, blur, smear or thin; frames, threshold, at, seconds, mix, depth — in freeze mode a modulator on port 1 scrubs the frozen frame through the sound), and resonator (choice the scale; key, spread, q, morph, rotate, bands, depth — ringing band-passes at the notes of the key, a modulator on port 1 rotating them through the scale). Sequencing, mixing and control (the second half of the rack, 2026-10-09): euclid replays the sound on a Euclidean pattern (steps, hits, rotate, bpm, rate, seconds, probability, accent, swing, gate, seed); turing pitches it from a looping shift register in a key (choice the scale; length, chance, range, key,

bpm, rate, gate, seconds, seed, transpose, rests); burst ratchets its hits (count, interval, bpm, rate, accel, decay, every, mix); mixer takes four sounds on ports 0–3 (gain1–4, pan1–4, mute1–4, gain, width); scan crossfades four sounds by position (depth, smooth; a modulator on its mod port moves it). The control cards take a modulation on port 0 and give a shaped one for any mod or partner port: quantize (choice the scale; key, range) snaps it to notes, sh (rate, or beats at the canvas tempo) samples and holds it, slew (rise, fall) eases it. The envelope's lpg choice is a low-pass gate (color, response); the reverb takes lowcut Hz and width; the synth card's choice drum:kick (or any drum voice) renders one hit of the drum machine. analyze (in the looking group) passes the sound through and reports key, tempo, LUFS and width on the card; tokey with shift 1 hears the sound's key and moves the whole sound to the target key. Note streams (2026-10-09): a sequencer card (its choice is the notes in the Sampler's grammar; bpm, rate, bars, probability, ratchet, swing, gate, velocity, accent, transpose, seed) makes a stream of notes rather than a sound; so do euclid and turing with midi 1, and followpitch turns a sung or whistled line into notes (choice chromatic or a scale with key; minMs, transpose, bpm). A stream goes into the synth card's port 0 (the preset plays it with up to eight voices), the multisample's port 0, or the sampler's notes port (1) — and that card into Output; the Output refuses a bare stream and says where it belongs. Every card keeps a lasting name: the id you gave it in the spec (“bass.filter”), or its kind and a number; nodes_get shows it beside the card’s position, and a knob or anything else that names the card still finds it after the patch is rebuilt or reordered. When the musician has ticked Agent patches wait for my OK in History, a new patch from an agent becomes a proposal instead, waiting to be auditioned and accepted. A patch that is refused changes nothing: the one before it stays on the canvas. A Source file over 1 GB is refused up front: a Source is held in memory, so trim or bounce the part you need first. x and y are optional, in canvas points (a card is 210 wide times the Text size, and a Sampler is twenty rows tall): cards given no place, or put within 32 points of each other, are laid out in columns by signal flow as Tidy does — the reply's notes say so, and nodes_get gives where each sits. A patch spaced by the caller keeps its places.

Ask Oscilla: nodes_get, nodes_plugin, nodes_set

## One piece back

Bring one piece back as a moment kept it — its cards and the wires that touch them — while every other card stays as it is now: the bass from an hour ago under tonight's vocal. The patch as it was is kept first, so the restore can be undone.

Ask Oscilla: piece_restore, pieces, checkpoint_list, checkpoint_restore

## Proposals

An agent's new patch can wait beside yours instead of replacing it. In File ▸ History… it is listed under Waiting for you with its note: Audition renders and plays it while your patch stays as it is, Accept makes it the live patch (yours kept as a checkpoint first), Reject sets it aside and keeps it in the record. Tick Agent patches wait for my OK and every agent patch comes this way.

Ask Oscilla: proposal_make, proposal_list, proposal_audition, nodes_set

## Set a proposal aside

The proposal is marked rejected and kept in the record; the live patch is untouched.

Ask Oscilla: proposal_reject

## Take a proposal

The proposal becomes the live patch; the patch before it is kept as an automatic checkpoint.

Ask Oscilla: proposal_accept, piece_restore

## The pieces of a song

A song is made of pieces you think of as one thing — the bass, the vox, the congas. A card named piece.role, such as bass.filter or vox.delay, belongs to its piece; a card without a dot belongs to main. Ask Oscilla which pieces the patch has and which cards are in each.

Ask Oscilla: pieces, piece_restore, nodes_set

## The proposals

The song's proposals newest first, each with its note, who made it, when, its cards and pieces, and whether it is waiting, accepted or rejected.

Ask Oscilla: proposal_list

## Tidy the Nodes canvas

Lay the Nodes cards out to be read: columns by signal depth — sources left, each card one column right of what feeds it, the Output last — with the rows inside a column ordered by where their inputs sit so wires run straight and cross little, cards stacked with their real heights, columns centered. The room’s Tidy button does the same; the canvas shows the result from the top left. nodes_set already lays out a patch whose cards were given no place or sit within 32 points of each other; Tidy is for a canvas that has drifted since. Cards sit 96 points apart across and 40 down.

Ask Oscilla: nodes_tidy, nodes_set, nodes_get

## What each stage of a chain changes

For the Nodes canvas as last rendered — source to output — or for files given in order, every stage against the one before it: the gain change over time in 50 ms windows (a compressor's or limiter's gain reduction shows as dips), the change in integrated loudness and crest, and the change per band, as one row per stage and as numbers. It finds the stage that squashed the groove.

Ask Oscilla: master_chain_trace, nodes_render

# Oscilla

Use help for instructions and status tools for live state; offline tools are a subset.

## A hardware instrument behind a lane

A hardware instrument behind a Tracks lane: a synth or groovebox on MIDI whose sound comes back on an audio input. Oscilla attaches a MIDI file and a device (with a channel) to a lane, or makes a new empty lane for it, armed. While the set plays, the file's notes play the device at the song's tempo from the lane's start, and every note is let go when the set stops or the loop goes round. Recording the lane — Record in the room or tracks_record — prints the device's sound onto it, placed earlier by the round trip, after which the notes rest until asked for again. Oscilla can measure that round trip: one note is sent and the input listened to, and the milliseconds are kept with the lane. Nothing is ever sent to a device that is not named.

Ask Oscilla: tracks_hardware, midi_search, tracks_record

## Agent note

Ask Oscilla to remember a lesson across sessions. A key makes it easier for another agent to find it.

Ask Oscilla: agent_note, session_note

## Agent notes

Ask Oscilla to recall saved lessons, optionally filtered by text or key.

Ask Oscilla: agent_notes

## Clear the knobs

Takes the knobs away; the panel's section stays where it is, empty, until the next knobs_set.

Ask Oscilla: knobs_clear, knobs_set

## Help

Search Help by task, room, control, or tool name. Open a result for instructions and related entries. This search does not search your sound library. Save as PDF… in the Help window writes the whole help as a book — the help of the build you are running, so it never drifts from the app.

Ask Oscilla: help

## Knob Panel

The Knob Panel: big labeled knobs for the sound you are working on, wired by Oscilla and turned by you. Ask for them in words — "give me knobs for this pad: swell, brightness, space" — and knobs_set puts 1 to 12 knobs in the Oscilla pane (the dial button in its header shows or folds them), each with a plain name, a range, a color and a line about it. Turn them while the sound plays: a knob on a Nodes card re-renders and loops the patch a moment after your hand stops; one on the synth or the drum machine is live — a synth knob turns the part that is playing, or the voice it names. A and B store two settings to flip between; Snapshots keeps up to eight named ones. When you have it right, Oscilla reads your values with knobs_get and bakes them in. The panel is saved with the project. A knob can also turn a knob on your hardware — a channel and controller number, or a named parameter on a device whose chart

the app knows. A knob can be turned from a MIDI controller too (right-click it for MIDI Learn, or map a row of the controller's knobs to follow the panel in Settings ▸ Controllers); while you are turning one there, a new knobs_set keeps your value for it. A knob can have a step — 1 for semitones, 5 for cents — so it clicks from one value to the next and reads in whole numbers; hold ⇧ while dragging any knob for fine moves. A knob aimed at a card by its position (n2) is kept by the card's lasting name, so rebuilding the patch does not move it onto another card.

Ask Oscilla: knobs_get, knobs_set, nodes_get

## Map a control

Maps a control on a device (a controller number, a fine-resolution pair, an NRPN, pitch bend, a note or program change) to a control in the app, or changes or removes a mapping. A knob can pick its control up when it reaches it, or jump; an endless encoder can send its turns in any of the three common ways. It can also make a row of the device's knobs follow the Knob Panel: knob 1 on the device turns the panel's first knob, whatever Oscilla wires there, and two buttons mapped to the next and previous page reach a panel with more knobs than the device has. And it switches automation writing on or off: with it on, a device's knob on a lane's volume, pan or send draws the lane's automation line at the playhead while Tracks plays, replacing what it passes over; the strip goes back where it stood when the hand lets go, one Undo takes the pass back, and the pass is listed among the hand changes.

Ask Oscilla: controllers_map, controllers_list, controllers_monitor

## MIDI clock to a device

Turns MIDI clock on for a device: while Tracks plays, the device gets 24 ticks a beat at the song's tempo, a start (or continue, with the song position) when playback starts and a stop when it stops, so its sequencer or arpeggiator plays in time. If its sound comes back late, move its clock earlier with an offset in milliseconds. It can also make Tracks follow a device's clock instead: that device's Start plays the set from the top, Continue from the song position it sends, Stop stops it, and the song's tempo follows its ticks.

Ask Oscilla: clock_set

## MIDI learn

Arms MIDI learn for one control in the app: the next control the person moves on any device is mapped to it. A few turns right after tell a knob from an endless encoder. The app shows 'Learned' over the window.

Ask Oscilla: controllers_learn, controllers_list

## One filed report (tool)

Ask about one report by its number: what it said, what was expected, when it was sent, and whether Iced Audio has fixed it — with the note that names the version to try. Only a report this Mac filed can be looked up.

Ask Oscilla: report_status, report_list, report_bug

## Oscilla

Oscilla is the app's assistant. Open the dock pane or use Ask Oscilla in a room. The room strips share one conversation. Send queues a request while work is running. The offline assistant handles browsing, playback, layering and simple edits; a connected model has the wider toolset. Oscilla needs a model of its own: download Oscilla Local (free, on this Mac) or

paste a key (a Claude sign-in appears only in builds that carry it). Separately, Help ▸ Connect Claude Desktop lets the Claude app drive AudioCortex from outside; when it or Claude Code is connected, the pane says so. With nothing connected at all, Oscilla answers with how to connect. The pane folds away with the button at its top right or View ▸ Hide Oscilla Pane (⌥⌘L), which also brings it back — useful when the Claude app or Claude Code is driving AudioCortex from outside and the room wants the width. The pane's header also holds the Song Map (the map button) and the Knob Panel (the dial button): big labeled knobs Oscilla wires for the sound you are on, which you turn by hand.

## Oscilla Local (the downloaded model)

Oscilla Local is Iced Audio's own small model for Oscilla, trained on this app's tools. It is a separate download with its own version number, updated on its own schedule in Settings ▸ AI ▸ Oscilla Local: Download, Update, Check Again, Remove. It runs entirely on this Mac, needs no key or account, and sits below a cloud key or a model server and above the on-device model. The file lives in Application Support, never in your documents. If a build has no local runtime, the card says so. The first time it answers on a Mac it reads the tool list once — a few minutes on older Macs, under a minute on recent ones — and keeps that reading on disk next to the model, so later launches load it in seconds; the AI tab shows this Mac's measured reading speed and the expected wait.

## Read the knobs

Oscilla reads the Knob Panel: every knob's value as you left it, which ones you turned since the last read, the A/B pair and your snapshots — so it can bake your settings into the song and learn your taste.

Ask Oscilla: knobs_get

## Report a Bug

Help ▸ Report a Bug… (also a link in the Help window). Say what happened and, if you like, what you expected and a way to reach you. The report goes to Iced Audio's own tracker, not by mail, and you get a report number back. Diagnostics go with it when the box is ticked: the app's version and build, macOS, the room in front, what was playing, and the app's last activity lines — tool names and timings, never your files or their contents. A screenshot of the main window goes only if you tick that box. A copy of what was sent, without the screenshot, is kept in Documents/AudioCortex/Bug Reports. Once a report is sent the window offers Close.

Ask Oscilla: report_bug

## Report a bug (tool)

Oscilla can file a bug report for you when something goes wrong: say what happened and it sends the report, with the app's diagnostics, to Iced Audio's tracker and tells you the report number. A description (the details) and steps (how to make it happen again) ride under the summary, so nothing a tester writes is dropped.

Ask Oscilla: report_bug

## See the controllers and mappings

Lists the MIDI devices AudioCortex hears, every mapping from a control on a device to a control in the app, whether each one's target is in the song now, and the tempo of a clock a device sends. With targets, it also lists every address a device can reach: Knob Panel knobs,

the synth, Tracks lanes the transport, and the Knob Panel's slots that follow whatever knobs are on it.

Ask Oscilla: controllers_list, controllers_map, controllers_learn

## Send to your hardware

Sends a controller change, a note or a program change to a MIDI device connected to the Mac — or a named parameter (a track's cutoff, its volume) on a device whose MIDI chart the app knows. SysEx is never sent.

Ask Oscilla: controllers_send, controllers_list

## Session note

Ask Oscilla to write a note about the current piece. The note belongs to this session's journal.

Ask Oscilla: agent_note, session_note

## Session status

Ask Oscilla what has been made this session. Captures retain their tool and argument history.

Ask Oscilla: session_status

## Stop all

Ask Oscilla to stop everything when you are unsure which engine is sounding.

Ask Oscilla: stop_all

## Watch what a device sends

The MIDI monitor: the last messages your devices sent — controller changes (with fine-resolution pairs and NRPN read as one value), pitch bend, program change, notes, start, stop, continue, song position and SysEx by its length. Clock ticks are summed into a tempo.

Ask Oscilla: controllers_monitor

## Welcome: Oscilla and requirements

Meet Oscilla shows the assistant’s current availability and connection choices. On a first launch it follows the 14-day trial’s sheet rather than arriving with it. The window leads with Oscilla Local — free, private, on this Mac — with its download and progress in place. “Let the Claude app drive AudioCortex” installs an extension in Claude Desktop so the Claude app can work AudioCortex from outside; that is a different thing from giving Oscilla a model here, and the pane says so when the extension is installed. Advanced opens Preferences for keys and model-server settings. Try Oscilla Now opens the conversation. The offline assistant needs a supported system with the on-device model ready. Connecting an assistant is separate from activating AudioCortex. When Apple's on-device model cannot answer something (a composing request, a long conversation, a busy or not-yet-ready model), Oscilla says so in plain words and points to Oscilla Local or a key — never an error code. The window also says plainly that AudioCortex is complete on its own: the Library, the instruments, every room and every engine run without any assistant — no account, no sign-in, nothing to install — on macOS 15 or later; Oscilla is an extra pair of hands, never a requirement (a reader had taken the window to mean the app needs Apple's AI or macOS 26).

## When the app froze

A freeze now leaves a report. The app keeps a heartbeat on its main thread; when it has not beaten for two seconds, the app samples itself with the system's own sample tool — this process only, nothing from any other app — into Documents/AudioCortex/Bug Reports as hang-<when>.txt, and when the main thread comes back the file says how long it was gone. Nothing is sent anywhere by itself: the newest hang report rides along in a bug report beside the crash report, and Oscilla can list them.

Ask Oscilla: hang_reports

## Your filed reports (tool)

Oscilla can list the bug reports filed from this Mac — by you in the Report a Bug window, or by an agent with report_bug — newest first, with each report's number, the date, what it said, and the tracker's word on it: open, fixed, won't fix or duplicate. A fixed report carries Iced Audio's note, which names the version the fix ships in, so you know when to try it again. The same list is in the Report a Bug window under Your reports. Only this Mac's reports are shown; nobody else's are readable.

Ask Oscilla: report_list, report_bug, report_status

# Sounds

sounds_browse shows choices; assign them with wavetable, synth_preset or fm_preset.

## Aurora presets

Aurora patches cover wide pads, brass, bass, strings and keys. Select one in Sounds to audition it, or ask Oscilla to apply it to a named voice. Each starts from a clean baseline. Shipped patches: Aurora Pad, Aurora Brass, Aurora Bass, Aurora FM Lead, Aurora PWM Strings, Aurora Sync Brass, Aurora Poly Keys, Aurora Noise Sweep.

Ask Oscilla: synth_param, synth_preset

## Mystical presets

Mystical patches are real bells, recorded by Iced Audio and played through the synth's sampler: a low Tibetan bowl, a higher one, a deep Tuu temple bell and a small hand bell. Each plays at pitch with its own strike and hum; the Ambient recipe's bells are the Tuu. Select one in Sounds to audition it, or ask Oscilla to apply it to a named voice by its full name. Shipped patches: Mystical Tibetan Bell, Mystical High Bell, Mystical Tuu Bell, Mystical Small Bell.

Ask Oscilla: synth_param, synth_preset

## Nova presets

Nova patches cover bright strings, brass, bass, sync and cross-modulation sounds. Select one in Sounds to audition it, or ask Oscilla to apply it to a named voice. Each starts from a clean baseline. Shipped patches: Nova Strings, Nova Brass, Nova Sync Lead, Nova Bass, Nova Sub, Nova Xmod Bell, Nova Pad, Nova Arp Pluck, Nova Hoover.

Ask Oscilla: synth_param, synth_preset

## Sounds

Browse wavetables, Nova and Aurora patches, FM patches and drum kits. Filter the browser, select a sound and audition it on the panel instrument. Putting that sound on an arranger voice is a separate assignment. A synth sound plays when you select it — click it or step with the Up and Down arrow keys, which work wherever the focus is except Oscilla's prompt (in the search field they step through what you found); the speaker button beside Audition turns that off. Drum kits wait for Load. A kit brings its sounds; its own pattern comes only to a machine that has none, so a beat that is playing keeps its steps. Play a sound when I click it turns the automatic audition on or off. Use in the Song puts the selected sound on one of the Arranger's voices. The keyboard along the bottom plays the sound that is loaded. The MIDI button at the right of that keyboard lets a MIDI keyboard play the selected sound, with its velocity; a drum kit plays from the General MIDI drum notes. Under the button is the name of the device the last note came from. MIDI is on by default; turning it on also turns on MIDI In in Settings if it was off. For a kit, Load onto the drum machine puts its sounds on the drum machine in Instruments; Play a beat runs the drum machine with it. The sources are hex-cut rows and the categories carry a mark in the sound color; each sound has a hex badge with its shape (a table's first frame) and its category as a chip; the loaded sound wears the sound color with an amber edge and three small bars; Audition is a hex-cut button in the sound color, Stop and the star are hexes; KEYBOARD PLAYS names the loaded sound above the keyboard.

Ask Oscilla: fm_preset, sounds_browse, synth_preset

## Sounds browse

Open Sounds to browse tables, analog patches, FM patches and kits. Select a sound to load the panel instrument; audition plays a short phrase.

Ask Oscilla: sounds_browse

# Spiral

## Spiral

The spiral shows live pitch energy from layered playback. Each turn is an octave; matching pitch classes line up around the circle. Play layered audio to feed it. It is a spectrum view, not a note transcription. It follows whatever is sounding: a sound auditioned in the Library, a song in the Arranger, the synth and drum machine, the DJ decks, layers and Tracks. The twelve directions are the twelve notes; each turn outward is an octave higher, and a bigger, brighter dot is a louder note. A legend along the bottom stays while music plays: direction is the note, outward is an octave higher, bigger and brighter is louder, and color follows the note, not the instrument. The rings are twelve-sided, the notes are small hexes with a halo on the loud ones, and the note labels are hex-cut chips that light in the note's color when it is loud. The twelve colors are the Appearance scheme's two blended round the wheel — C is the sound color, F♯ the motion color — shown in a wheel at the top right, so the Spiral follows the scheme too.

Ask Oscilla: detect_key, play_layers

# Synthesizer Sampler

## Connect the sampler

Choose the MIDI output and channel the synth listens on, the audio input (an empty name means the input chosen in Settings ▸ Audio ▸ Audio Input, else the Mac's default) and the channels it comes back on ([0, 1] a stereo pair, [2] one channel mono), the synth and patch names that become the library's folders, and a program change to send first. The input opens and reports its sample rate and channels. Every field is optional; what is given changes. library_root moves the libraries to a folder of your own (made if missing; a synth folder and a patch folder inside it, as before); '' goes back to Documents/AudioCortex/Sample Libraries/instruments. The room's Library row shows it, with Choose… and Default. patches fills the room's Patches field (the walk Start will run) without starting it; sampler_run with patches walks at once.

Ask Oscilla: sampler_devices, sampler_setup, sampler_test_note, sampler_run

## One note, read back

Play one note through the setup and read what came back: the peak in dBFS, how long it rang, how many milliseconds after the note-on the sound arrived, the pitch heard and whether it matches the note asked for, and plain warnings — silence, clipping, too quiet, late, an octave off, out of tune. The take is written to Recordings so it can be heard.

Ask Oscilla: sampler_plan, sampler_run, sampler_setup, sampler_test_note

## Record the library

Start the run: every note in the plan at every velocity, each take trimmed to the sound, faded in its last milliseconds, and written as a 24-bit WAV into Sample Libraries ▸ instruments ▸ <synth> ▸ <patch>; at the end a <patch>.json manifest and a <patch>.sfz. Returns at once (wait: seconds holds the reply up to that long); sampler_status follows it. {cancel: true} stops after the take in flight — what was taken stays. Three silent takes with nothing taken stop the run by themselves. A WALK samples a whole synth in one go: give patches as text — a range of programs (0-127), a list (5, 9, 12), or a bank before a colon (1.0:0-63, bank MSB.LSB) — and every patch is sent as bank select and program change, sampled with the same plan into its own folder (named from a pasted names list, else by its number), with a settle pause between patches; the status counts patch n of N and the room shows the same under its Patches field.

Ask Oscilla: sampler_library, sampler_plan, sampler_run, sampler_status

## Synthesizer Sampler

Record a hardware synthesizer into a sample library. Plug the synth's MIDI in to the Mac and its audio out to an input, choose both here, name the synth and the patch, and set the plan: which notes (from, to, every n semitones), which velocities, how long each key is held and how long to wait for the release. A test note shows what comes back — its level, how late it arrives, and whether the pitch is the one asked for — before the run. The run plays every note at every velocity, trims each take to the sound, and writes 24-bit WAVs into Documents ▸ AudioCortex ▸ Sample Libraries ▸ instruments ▸ <synth> ▸ <patch>, with a manifest (every take's note, velocity, length, peak, pitch check) and an SFZ that maps the takes across the keyboard. The app's own synth is a destination too — 'AudioCortex Synth' — so the room can be learned without a cable; any app on the Mac can play that destination as well. Stop ends a run after the take it is on; what was taken stays. The Patches field under the patch name walks a whole

synth: a range or list of programs, a bank before a colon, each sampled into its own folder in turn. The first time the sampler opens an input, AudioCortex asks to use the microphone; allow it and open the input again.

Ask Oscilla: open_window, sampler_devices, sampler_library, sampler_plan, sampler_run, sampler_setup, sampler_status, sampler_test_note, synth_preset

## The libraries made

Every <synth>/<patch> folder under Sample Libraries ▸ instruments with its sample count, note range, velocity layers, when it was made, and the manifest and SFZ paths. synth narrows to one synth; patch to one patch, whose manifest comes back in full.

Ask Oscilla: sampler_library, sampler_run

## What the run will record

The grid: low, high and step (midi notes, semitones between samples — C2 to C6 every three by default), or an explicit list of notes; the velocities (one layer by default; [40, 80, 127] for three); hold (seconds the key is down), tail (the longest wait for the release), silence_db (the line under which the tail is over), round_robins (takes per note and velocity). Returns the grid and how long it should take.

Ask Oscilla: sampler_plan, sampler_run, sampler_test_note

## What the sampler can connect to

Every MIDI output on the Mac — the hardware synth's interface among them, and 'AudioCortex Synth', the app's own synth as a destination — every audio input with its channel count, and the sampler's current setup, plan and state. The place to start.

Ask Oscilla: sampler_devices, sampler_setup

## Where the run is

The state (idle, testing, running, done, stopped, failed), done of total, the note it is on, elapsed and remaining seconds, takes so far, the last failures and warnings, the folder, the manifest when done, the last test note's reading, and how many notes the app's own synth has been sent over MIDI.

Ask Oscilla: sampler_run, sampler_status

## Who is the synth

Ask Oscilla who the synth on the sampler's MIDI output is: a Universal Device Inquiry goes out and the reply — the maker, named from the common manufacturer ids, and the maker's own family, member and version numbers — comes back on the Mac's MIDI inputs within the seconds given. The synth's MIDI out must reach the Mac for a reply; the app's own synth answers no inquiry, being no device. Patch names by SysEx are per model and come after this, synth by synth.

Ask Oscilla: sampler_identify, sampler_devices, sampler_setup

# The Explorer

## Explorer configure

In the Explorer, change the center style, move through categories or turn the page. Ask Oscilla to read the current view before navigating.

Ask Oscilla: explorer_configure

## Explorer show

Open the Explorer to explore the Library as a category tree, similarity constellation or playable harmonic table.

Ask Oscilla: explorer_show

## The Explorer

The Explorer opens in its own window. Explore the loaded list as a category tree, orbit similar sounds around a reference, or turn a sound into a playable harmonic table. Taxonomy mode analyzes an unclassified list first. Its panel follows the main window's list: with sounds listed but none analyzed it offers Analyze, and once they are analyzed the tree builds. When some are left that cannot be read as sound (damaged, empty, or a format the classifier cannot open), the panel says so under its Analyze button rather than offering them again in silence. Opening a folder after the window is up is enough; it looks again whenever it comes forward. With the tree up and nothing played yet, the pane says what a click does, what Play, Stop and Record do, and what the looks and the constellation are; once a sound is played the pane holds the list of what was played. In the Hive look a faint hex lattice sits behind the tree; a pitched branch is edged in the sound color and a struck one — percussion, one-shots, hits, effects — in the motion color, with pips under its count for how full it is; Send is amber, LOOK is a hex-cut chip that opens the center styles, and Play, Stop and Record are hexes.

Ask Oscilla: explorer_configure, explorer_show

# Tone Finder

## Tone Finder

Enter a frequency in Hz. The tool shows the nearest note, MIDI number, cents offset and the exact frequency of that note. It does not listen to a microphone.

# Tracks

Inspect the assembly before replacement; stems_reinforce adds parts and project_bounce keeps a project render.

## A lane’s fader and pan (tool)

A lane’s fader in dB and its pan, as the strip’s knobs set them. Not undoable, like the knobs; the playing set follows at once.

Ask Oscilla: tracks_fader, tracks_mix, tracks_lane, tracks_clip

## A lane’s send (tool)

How much of a lane goes to a return lane, and from where: a post-fader send (the default) rides the lane’s fader, as the built-in reverb and delay sends do; a pre-fader send is taken before the fader, so the return still hears the lane when its fader is down. Mute silences both. tracks_mix shows each lane’s return sends beside the built-in two.

Ask Oscilla: tracks_send, tracks_return, tracks_mix

## A mix onto the lanes

A kept mix back onto the lanes, by lane name, while the song plays — faders, pans, sends, mutes, solos, inserts and the master move to it at once, so two mixes can be heard against each other with the music running. Lanes the mix does not name are left as they are.

Ask Oscilla: tracks_snapshot_load

## A mix, kept by name

Keep the mix as it stands — faders, pans, sends, mutes, solos, inserts and the master — by name, in the song's Mixes folder: the agent's mix, then yours, then yours v2. In the room, the ⋯ menu's Save Mix… does the same. Mixes switch while the song plays and compare lane by lane.

Ask Oscilla: tracks_snapshot

## A take back on its lane

Ask Oscilla to put an earlier take back — take 1 after take 3 replaced it. It goes on the lane named, else the lane it was recorded on, else a new lane, at the place it was played. Undo in Tracks brings the previous audio back.

Ask Oscilla: tracks_take_use, tracks_takes

## A track's structure, from the file

The same reading as listening, from a file: the tempo with where bar 1 falls, the key, and the sections on the bar grid — a long dance track as its sub-drops, kickless plateaus, false returns and breakdowns.

Ask Oscilla: analyze_structure, section_map, spectrogram_image, loudness_timeline_image

## Act on whole lanes

Ask Oscilla to duplicate lanes (each copy right under its original, with the same file, clips, knobs, sends, effects, automation and color), remove them, mute or unmute them, move every

clip of several lanes together by a number of seconds, or consolidate them — each lane's clips, fades and gain and pan lines rendered into one file in the project's Stems folder (send and effect lines stay on the lane). Every action is undoable, as with lanes picked by hand in the room.

Ask Oscilla: tracks_lane, tracks_clips, tracks_undo

### Analyze mix

Ask Oscilla to analyze stems together before mixing. It reports loudness, pitch and spectral properties, and where pairs mask one another. Give from and to in seconds to measure only that passage.

Ask Oscilla: analyze_mix

### Automate a project lane

Ask Oscilla to put a gain or pan line on a project's lane; the project's bounce renders it and project_tracks shows it. The points' t is seconds on the project timeline, not from the lane's start — a lane placed at 490 s takes points at 490 and later.

Ask Oscilla: project_automate, project_tracks

### Balance start

Ask Oscilla for a starting balance of stems. It measures loudness and loads equal-loudness gains into Tracks; use the faders to make the musical balance.

Ask Oscilla: balance_start

### Bounce the Tracks room

Ask Oscilla to bounce the Tracks room's lanes — clips, gains, pans, sends, inserts, every automation line and the master — to a WAV, as the room's Bounce button does, without a dialog: when the room shows a song, into the song's Renders folder as its next numbered version, kept on the song's record like a project bounce; a loose set's bounce goes to Recordings. `overwrite` true lets it replace a file that already exists at `path` (otherwise an existing file, and any lane's own file, is refused).

Ask Oscilla: tracks_bounce, tracks_clips

### Change a lane

Ask Oscilla to change a lane of a project in place: its level, its place on the timeline, its file, mute, or its name — nothing is removed and no stem is trashed. A muted lane stays in the arrangement and is left out of the bounce. It can also set a lane's pan and its reverb and delay sends, or solo it; the Tracks window shows the change at once.

Ask Oscilla: project_set_track, project_tracks

### Channels and plugins

The chip at the top of a strip is the lane’s number and name in its color: double-click it to rename the lane, right-click it for a color (a palette, any color from the panel, or automatic). Use a lane’s gain and pan to place it in the mix. M mutes; S solos. Sends feed reverb and tempo-based delay. Right-click the gain, pan or a send knob for MIDI Learn, then move a control on your MIDI controller to drive it. Insert chips open controls; their menu bypasses, reorders or removes them. Add built-in effects or installed Audio Units, and rescan to refresh

that list. The master has inserts, gain and an optional limiter with a ceiling. Among the inserts, Ember EQ (Iced Audio: Ember EQ in the picker) is the app’s own character EQ, for every lane and the master. An Audio Unit that cannot load passes audio through and reports the problem. Save Tracks retains the assembly and plugin state. Right-click a lane's name for Duck Under: the lane dips while the lane you pick sounds, after its inserts and before its fader; the strip shows DUCK with that lane's name.

Ask Oscilla: tracks_duck

## Checkpoints: keep a moment of the song

When the song is somewhere you love, File ▸ Checkpoint… (⌥⌘S) keeps it exactly as it is: the open project, the Nodes patch, the Knob Panel and the Tracks lanes, plus a copy of any of your own audio files the lanes play. Give it a short name and, if you like, a note. A checkpoint takes almost no disk space until something changes, and it is read-only, so nothing that happens later can spoil it. File ▸ Moments… lists them: play a moment's latest render without restoring it, show it in Finder, or Restore it. Restoring always keeps the song as it is first, as an automatic checkpoint, so a restore can be undone; nothing is deleted. Before an assistant replaces the Nodes patch, the patch and knobs are kept automatically too. Your assistant can make and restore checkpoints as well — the app keeps them, so they are there whichever assistant you use. Checkpoints live inside the project (in its Checkpoints folder), or in Documents/AudioCortex/Checkpoints when no project is open. Tick Lock it in the Checkpoint window when the moment is a decision (“that’s the bass”): a lock is shown with a padlock in Moments and written into the song’s history, and the phone remote’s Song Map panel has a Lock it button too. A project also keeps its Nodes patch inside it now (Nodes/patch.json), so opening the project brings its patch back, and your own changes — a card’s setting, a knob, a lane’s fader — are written to the history beside the assistant’s and the phone’s. Two are automatic: one at quit when the song changed since its last checkpoint, and one a day while a song stays open and changing; both show under Show automatic in Moments. The Song Map, the line saying what Oscilla is working on, and the loop are kept too.

Ask Oscilla: checkpoint

## Clickiness

Ask Oscilla how clicky a take is — field recordings, hardware, a dropped cable: the De-click card's detector counting only. Clicks, clicks per minute, the first fifty with their times and how far above the signal's own level each stood, the five worst, and a verdict — clean under two a minute, some under fifteen, clicky above.

Ask Oscilla: clickiness

## Clips on the lanes

Ask Oscilla what clips the Tracks lanes hold: where each starts on the timeline, which piece of its file it plays, its gain and fades. A lane that has never been edited shows one clip, the whole file. Each lane also reports its own level in dB, pan, mute and solo, so a change made by hand or on the phone is visible. A lane that carries notes says so: how many, their sound, and whether their render is still being made.

Ask Oscilla: tracks_clips

## Collect all files into the song

Tracks ▸ ⋯ ▸ Collect All Files into the Song (or ask Oscilla) copies every sound a song plays from outside it — your sample folders, a DAW's temporary project, the Desktop — into the

song's Stems folder, and the lanes play the copies from then on. Your originals stay exactly where they are. Collect again and nothing is copied twice. Files dropped onto Tracks from a folder that clears itself (a DAW's temporary project, the temp and cache folders, the Trash) are kept as a copy in AudioCortex/Recordings/Kept straight away, so nothing goes with the folder.

Ask Oscilla: project_collect

## Compare two files

Ask Oscilla to compare two files — a piece against a reference, this bounce against the last. It reports each one's level, crest, brightness and width with the deltas, the eight-band balance of each and the shape difference with loudness removed (brighter or darker, not just louder), the moments where they diverge most, and the bands where both are loud at once. Give from and to for the first file and b_from and b_to for the second to compare a chorus with a chorus.

Ask Oscilla: compare

## Concat files

Ask Oscilla to join audio files end to end into one stereo WAV in Recordings — several section bounces become one longer section. When any of the files is a float file, as a Nodes render is, the joined file is float too, so nothing is squeezed into 16 bits. An optional crossfade overlaps the joins. The reply says where each part starts and the result's peak and RMS.

Ask Oscilla: concat_files

## Do the stems line up?

Ask Oscilla whether your stems line up at a moment — a downbeat or a hit they all play. It finds where each stem's first attack begins (where its sound rises, not where it is loudest, so an echo is not mistaken for the hit) and says which stems are early or late against the first one, in milliseconds, and how far to move each.

Ask Oscilla: stem_alignment, beat_grid, onset_grid_image

## Draw an automation line

Ask Oscilla to draw a line on a Tracks lane: gain (dB), pan, the reverb or delay send (rev, dly, 0…1), or one setting of an effect in the lane's insert slots (fx1:cutoff — the slot from 1 and the setting's name). Points by time, straight between them, held beyond the ends. Gain and pan lines sit on top of the strip's knobs; a send or effect line takes over its knob while the set plays and in the bounce. Undoable. A point can carry a curve, –1 to 1: the shape of the line to the next point — 0 straight, towards 1 the value holds and moves late, towards –1 it moves early and eases in. In the room, drag over a line to draw it with the pencil, ⇧-drag for a straight line, ⌥-drag a segment to bend it; right-click a segment to bend it late or early, or straighten it.

Ask Oscilla: tracks_automate, tracks_automation, tracks_clips, tracks_undo

## Drop files onto a track

Drag audio files from the Finder or the Library onto a lane and they land there, at the time under the mouse (on the grid; hold ⌘ to place freely). Several files go one after another. A lane outlines in blue while a file is over it. On a lane that already has audio, a dropped file becomes a clip over it: what lies under the new clip is cut away and the rest plays on — nothing is deleted, the files stay where they are, and Undo puts the lane back. An empty lane (+ Track) takes the first file as its audio. Drop files anywhere else in the room, or on an empty room, and

each becomes a new lane. Oscilla can place files the same way.

Ask Oscilla: tracks_place, tracks_undo, tracks_clips

## Ducking between lanes

Sidechain ducking: a lane dips while another sounds — a bass under a kick, a pad under a voice. Right-click a lane's name and choose Duck Under, then the lane to listen to; the strip shows DUCK with that lane's name, and the same menu sets how deep it dips (gentle 3 dB to hard 24 dB) or stops it. Oscilla sets it with the amount, the attack and release it follows with, and the source level the dip begins at. The dip sits after the lane's inserts, before its fader and sends, so it is heard live and in a bounce and in the lane's stem alike.

Ask Oscilla: tracks_duck

## Edit a clip

Ask Oscilla to split a clip at a time, move it, trim its edges, fade it, change its gain, delete or duplicate it. Every edit is undoable and the room shows it; Play and Bounce render the result. A clip can also be moved to another lane (to_lane), or copied there (copy) leaving the original, as a person does by dragging it (⌥ for a copy). Reverse plays the clip backwards from a copy of its own, leaving the file it came from untouched; right-click a clip for Reverse Clip, or Edit in Editor to open its file. Warp stretches what a clip plays to the song's tempo (or a tempo given) with the warp engine — beats for drums and anything with hits, complex for mixed material, texture for pads — into a file of its own next to the song, and the clip plays that from where it starts; a clip after it gives way, the original file is untouched, and Undo brings the clip back. The clip's menu has it as Warp to Tempo. The clip's own tempo is read from it unless source_bpm gives it; every beat of the clip lands on the same beat of the song's grid. Groove lays a feel onto a clip as it is warped: from a file whose groove is read as the groove map reads it — where its hits sit against each sixteenth — or a plain swing by percent (50 straight, 58 a light shuffle, 66 triplets), scaled by strength. The clip is taken as straight at its own tempo and every sixteenth lands where the groove puts it, at the song's tempo. The clip's menu has Apply Groove with a file and the usual swings.

Ask Oscilla: tracks_clip, tracks_clips, tracks_undo

## Ember EQ: the character EQ for every lane, the master and the canvas

Ember EQ is the EQ the classic passive and inductor moves are made of, as one unit: a steep high-pass (off, 50, 80, 160 or 300 Hz, 18 dB per octave); a low band at 20, 30, 60 or 100 Hz with Boost and Atten on separate knobs — the boost is a wide, gentle shelf and the attenuation a cut centred above it, so both together lift the lows and scoop the low-mids, the trick the originals are used for; two broad inductor style bells (220/360/700 Hz and 1.6/3.2/4.8/7.2 kHz, ±18 dB, narrower as they are pushed); a high boost at 5, 8, 10, 12 or 16 kHz with a bandwidth knob and a separate gentle attenuation shelf at 5, 10 or 20 kHz; a Drive stage (soft second and third harmonics, a little low bloom as it is pushed, oversampled so nothing folds back) and an output trim. Flat with Drive at 0 it is unity gain, and bypass is click-free. It is in three places: the FX picker of every Tracks lane and the master (Iced Audio: Ember EQ, with its knobs on the chip’s editor, saved with the take and automatable like any insert), the Nodes canvas (the Ember EQ card, in the tone family, with a readout naming what is moved), and as an AudioUnit any card that hosts units can take by kind au:aufx:embr:IcdA. Its curves are fitted to measurements of the hardware-modelled units they stand in for, so a move lands where your ears expect it. The Live export does not carry it (Live cannot load the app’s own unit): bounce the lane, or make the move again in Live.

Ask Oscilla: tracks_lane, tracks_return, nodes_set

## Ensemble

Ask Oscilla to rebuild a target as layered library sounds. The selected stack is bounced and loaded into Tracks for adjustment.

Ask Oscilla: ensemble

## Export to Ableton Live

Ask Oscilla to make an Ableton Live project from a song. You get a folder with a Live set and every sound collected beside it: each lane is a Live track with its clips where they sit, its fades, level, pan and sends, plus the tempo, the Song Map sections as locators (the room's map as it stands when the room is exported), and the song's MIDI parts. It needs Ableton Live on this Mac. If some of the song's sounds can't be found — say its folder was moved — the export lists them and says how to point it at the new place. The room’s return lanes go out as return tracks after A and B, with every lane’s send on them; group lanes are not written yet (their members go out as plain tracks).

Ask Oscilla: export_ableton, tracks_return, tracks_group

## Glitch finder

Ask Oscilla for the hiccups: broadband spectral discontinuities with no matching level change — a stem switch, a dropout, a seam, a click under the music — the kind of thing only a vertical line on a zoomed spectrogram showed on The Empath's Path at 0:56. Each one comes with its time, how many times the local spectral flux it is, the percentage of bins that jumped, the level change, and whether the spectrum came straight back (a line, not a new section). A musical hit moves the level too and is not listed.

Ask Oscilla: glitch_finder

## Group lanes (tool)

A group lane is a folder: its member lanes sum into the group’s bus, through the group’s inserts, fader, pan and mute, into the parent group or the main mix. Solo on a group solos its members; groups nest. tracks_group lists, adds, removes and sets them, and moves lanes in and out; ungrouping puts the members back where the group sat. The playing set rebuilds as needed and a bounce carries the groups.

Ask Oscilla: tracks_group, tracks_mix, tracks_lane, tracks_return

## Hide the Sounds tab

Back to the Map tab; the palette is kept and any clip playing stops.

Ask Oscilla: sound_palette_hide

## Import a Live set (tool)

An Ableton Live set comes into the Tracks room: the Tracks menu’s Import Ableton Live Set…, or Oscilla by the .als path. Every audio track becomes a lane named after it, with its fader, pan, the two sends and mute on the strip; each clip in the arrangement lands where it sits with its trim, fades and clip gain; the set’s tempo becomes the room’s and its locators the Song Map’s sections. Samples are found by the set’s own paths, else by name under the set’s folder; one not found is named. A warped clip is laid at its first warp ratio, since the room does not stretch;

The set's group tracks become group lanes with their members, and its return tracks beyond A and B become return lanes with each track's send; MIDI tracks are counted, not brought in. Sets from Live 10, 11 and 12 read.

Ask Oscilla: import_ableton, export_ableton, tracks_clips, song_map_set, tracks_return, tracks_group

## Import an 0RKESTRACKER song (tool)

A song born in 0RKESTRACKER comes into the Tracks room from its stem export: in 0RK, Export ▸ Stems writes one WAV per channel and a manifest beside them. Import that folder here (the Tracks menu's Import 0RKESTRACKER Stems…, or Oscilla by its path) and every channel becomes a lane named after it, with its volume, pan and sends on the strip; the tempo is set; the pattern order becomes the Song Map's sections; the room takes the song's title. Import the same song again after a change in 0RK and each lane's audio is replaced while the faders, pans, sends and inserts you set here stay — the mix survives the round trip.

Ask Oscilla: tracks_import_ork, tracks_place, tracks_tempo, song_map_set

## Lane height

Make the lanes taller or shorter with ⌥-scroll over them; ⌘-scroll zooms time around the pointer, or around the selected clips when some are picked; the zoom buttons and slider keep the selected clips in view, else the playhead or the cursor where they sit on screen; when neither is on screen, the middle of the view stays put. Below the full height the channel strip folds to the name, its buttons and the level, and under 120 points to the name and level. The height is kept between launches. Oscilla can read or set it.

Ask Oscilla: tracks_view

## Level waves

Ask Oscilla to measure the volume waves through a transition you can hear: the level in 50 ms steps with its 3 s trend removed, and every 4 s window whose ripple swings more than the minimum (1.5 dB) reported as a wave with its depth and its period from the ripple's autocorrelation, plus the deepest ripple anywhere and where it is. A steady crescendo is trend and is not a wave; a pumping compressor or two layers beating against each other is.

Ask Oscilla: level_waves

## Listen to anything playing

To study how a favorite track is built — to measure it and understand it, never to copy it — Oscilla can listen to whatever plays into an input: a track in a browser sent into BlackHole through a Multi-Output Device, an interface, a microphone. Every half second it keeps numbers only — loudness, six band levels, brightness, flux and the twelve pitch classes; when you stop, the tempo, the key, where bar 1 falls and the sections on the bar grid. The sound is measured in memory and let go: nothing is recorded, played or sent. If the input stays silent it says why — nothing routed into BlackHole, a browser still on its old output, the input and output at different sample rates. The first time an input is opened, AudioCortex asks to use the microphone; allow it and listen again.

Ask Oscilla: listen_start, listen_status, listen_stop

## Listening, now

While listening: the input, how long, the latest half second of numbers, and after three seconds of nothing the reason the input is silent and how to fix it.

Ask Oscilla: listen_status, listen_start

## Listing the song's checkpoints

The song's checkpoints, newest first, with who made each one and what it kept. Automatic ones (before a restore, before an assistant replaced the Nodes patch) are hidden unless asked for, and the list says how many it hid; the newest twenty automatic ones are kept and named ones are never removed. With no song open it reads the song shown in Tracks, or any song you name.

Ask Oscilla: checkpoint_list

## Locking a decision

A lock marks a piece of the song you have decided on. It keeps the whole song as it is, like a checkpoint, and says in the song's history what was decided, so you, your assistant or a later session can see how the song got here and go back to that moment.

Ask Oscilla: lock

## Loop a region

The Loop button (⌘L) loops the set; ⇧-drag in the time ruler draws a loop brace and turns the loop on, so only that stretch repeats. ⌘L with a clip selected makes the clip the loop. Right-click the ruler to clear the brace — then the whole set loops. Every lane goes round at the same place, and a playing set changes at once. Oscilla can set, read and clear the loop the same way.

Ask Oscilla: tracks_loop, play_layers

## Master and release

Ask Oscilla for the release package of a mix in one go: a linear master — one gain toward a loudness target (linear –14 LUFS, loud –9, quiet –18, or a target of your own), a gentle tilt if you ask, a true-peak limiter under the ceiling, so the sections keep their shape — then the files: the 24-bit 48 kHz master, a dithered 16-bit 44.1 kHz copy, a tagged AAC for phones, the master muxed into a music video if you give one, and release.json with the loudness, the true peak, the length and the list. It never shortens the piece. Files land in Documents ▸ AudioCortex ▸ Releases ▸ <name> unless you say where. Give a mix file, or a project to take its latest render — the release is then noted in the project's journal.

Ask Oscilla: master_and_release, project_bounce, loudness

## Missing files in Tracks

A lane whose file is gone — a drive unplugged, a folder moved — no longer stops Play: the lane plays silent, its name turns red with a missing file badge, the status line names it, and the rest of the set plays. A bounce and the stems still refuse until the file is found, because a render with a hole in it is the worse surprise. Find Missing Files… in the ⋯ menu asks for a folder and looks through it, folders inside it too, for each missing file by its name, then points the lanes and clips at what it finds; one Undo puts the old paths back. Oscilla lists what is missing and looks for it the same way.

Ask Oscilla: tracks_missing, tracks_play

## Mix files

Ask Oscilla to mix files with per-file gains, and with start offsets in seconds to place each one on a timeline. It writes one stereo WAV in Recordings, pads shorter files with silence, resamples mixed rates, and reports the result's peak and RMS.

Ask Oscilla: mix_files

## Mix map of a project

A Tracks project as a heat map: one row per lane, color = the lane's level over time after its gain, offset and mute, names at the left, section markers optional. Per section it lists which lanes are heard (over –40 dBFS) and which are silent — overlaps, doubled lanes and empty slots at a glance.

Ask Oscilla: mix_map_image, project_list, project_tracks

## Name a lane or the set

Name a lane (lane, name) and the set (title) in the Tracks room, and give a lane its color (color: a palette name — indigo, steel, sky, gold, violet, green, coral, silver — any #RRGGBB, or auto for the cycle by lane number). The name is in the strip and the badge, the color on the badge, the clip and the knobs; both are saved with the set. With no arguments it just lists the names and colors.

Ask Oscilla: tracks_rename, tracks_clips, project_set_track

## Notes on a lane

A Tracks lane can carry notes — a MIDI clip — and a sound to play them: an analog preset, an FM preset or a wavetable from the library, or the plain synth; notes on channel 10 go to the drum machine. What the lane plays is those notes rendered by the app's own synths into a file of its own, next to the song, so play, bounce, inserts, automation and the Live export see an audio lane as before. Oscilla puts notes on a lane from a list or from a MIDI file, adds, removes, moves, transposes and quantizes them, and changes the sound; the strip's N button opens the piano roll over the lane: click to add a note on the grid, drag to move it, its right edge to stretch it, double-click to remove, right-click for velocity, octaves and quantize. When the notes change the lane is rendered again in the background and the old file stays, so Undo brings the notes and the sound back. Unchanged notes are never rendered twice.

Ask Oscilla: tracks_notes

## Play layers

Ask Oscilla to layer up to eight files. Each plays without sampler filters or envelopes. Loop repeats until stopped; mix the files to keep the blend.

Ask Oscilla: mix_files, play_layers

## Play the Tracks room

Space plays and stops the set; a click in the ruler moves the cue, where Play starts. Oscilla can do the same: play from the cue or from a time, pause (the cue stays where it stopped), stop, or move the cue — a playing set carries on from there. The master's loudness comes with every reply, and action loudness asks for it alone: momentary, short-term and integrated LUFS since the play began, and the true peak held — the same reading the master strip shows under its meters; reset starts the integrated reading again. A lane whose file is gone plays silent and the reply names it under missing.

Ask Oscilla: tracks_play, play_layers

## Project remove track

Ask Oscilla to take a lane out of a project's arrangement by name. A stem the project owns goes to the Trash with it; a referenced file is left where it is.

Ask Oscilla: project_remove_track, project_add_track

## Record in Tracks

Record works like recording onto one track in a DAW. Arm a lane with R on its strip (one at a time), or press + Track for a new empty lane that is armed already; then press the red Record button beside Stop. The set plays from the cursor while the audio input — the one Settings ▸ Audio ▸ Audio Input names, otherwise the Mac's default — records; the status line under the lanes shows REC, the time, the input level and the input's name. Stop (or Record again) ends the take: it is saved as a WAV in Documents/AudioCortex/Recordings and goes onto the armed lane at the place you played it, keeping the lane's name, knobs and effects. Every take is a new file — nothing is overwritten. On a lane that already has audio the take is punched in: it plays over the part you recorded and the rest of the lane plays on; Undo takes it back out. With no lane armed the take lands as a new lane, Take 1, Take 2 …; in an empty room it simply records from the cursor. The metronome button beside the tempo clicks at the set's tempo while it plays and records, and COUNT beside it steps the count-in through none, 1, 2 and 4 bars. Monitoring sits on the armed lane's strip beside A and P: Off (never), Auto (heard while recording — the default) or In (heard whenever the lane is armed); use headphones. If a take sits a hair off the beat, select it and nudge it with ⌥ ← →. Oscilla can do all of this too. The first time, AudioCortex asks to use the microphone (macOS gates every audio input, an interface as much as a mic, behind that one question); allow it in the dialog and record again. If it has been refused, Record says so and where to allow it: System Settings ▸ Privacy & Security ▸ Microphone.

Ask Oscilla: tracks_record

## Reinforce a stem pack

Drop a stem-pack folder or zip into Tracks. Open Reinforce and check the Bass, Drums and Chords source labels. Enable Sub, Kick+ and Snare+, Hat+, Hook or Pad and choose presets. Pattern mode follows recurring hits rather than every onset; Hits moves the added hits earlier or later in milliseconds. Hook composes a motif using the key and chords; Pad doubles the chord chart. Reinforce renders new lanes beside the originals. Rerunning replaces the earlier generated lanes without overwriting their old audio files.

Ask Oscilla: stems_reinforce

## Render a song from a score

Ask Oscilla to build a whole song from a score: a title and tempo, then layers — each a file with a start in bars (or seconds), a trim, a gain, a pan, reverb and delay sends, a gain envelope drawn at bars, and, when the sound needs shaping, a Nodes chain with {SRC} standing for the layer — plus the master (target loudness, ceiling, maximizer, reverb, delay). Every layer becomes a lane with its line in the project, the mix is bounced, and a processed stem per layer is written at its place on the timeline (Stems ▸ Processed). The score itself is kept in the project (Score ▸ score.json) so a re-run is one line; a second run with nothing changed is quick, because trims and chains come from their caches. It never shortens a song; a layer whose file is missing is named, a muted one left out and named. A changed score for the same project replaces the lanes of the one before: lanes the last score named that this one does not come out (the reply's removed names them), a lane of the same name is written over, and

lanes added by hand stay; renders and the journal keep their history.

Ask Oscilla: project_render_score, project_bounce, project_tracks, master_and_release, nodes_set

### render_cancel — stop a long render

Stops a long render: a bounce, the stems, a clip warp or a note-lane render. The render ends at its next block or lane; the call that started it returns an error saying it was stopped, and nothing half-made is kept where its result would be — a stopped bounce writes no file, a stopped stems export keeps the stems already complete and no partial one. In the Tracks room the same is the Stop button beside “bouncing…”, and Stop All stops every render too.

Ask Oscilla: render_cancel

### render_jobs — the long renders running now

Lists the long renders running at this moment — a bounce, the stems, a clip warp, a note-lane render — with how many seconds each has run and whether it is already stopping. A Nodes render and an arranger record are followed by their own tools (nodes_render with cancel, job_status).

Ask Oscilla: job_status, nodes_render, render_cancel, render_jobs, song_stop, tracks_bounce, tracks_stems

### Restoring a checkpoint

Restore puts the song back as a checkpoint kept it — all of it, or just one part such as the Nodes patch. The song as it is first becomes an automatic checkpoint, so you can always go back again. If one of your own audio files changed or went missing since, the lane plays the copy the checkpoint kept; your file is left alone. With lane — a lane's name or index — only that lane comes back as the moment kept it (level, pan, mute, solo, clips, automation) and the rest of the song is untouched; the arrangement as it was is kept first. Name a checkpoint by its id or its whole label, however long; a beginning that fits only one also works.

Ask Oscilla: checkpoint_restore

### Retitle a project

Ask Oscilla to give a project a new title. The package keeps its folder, so paths held by other tools stay valid; the title is what the app lists and shows.

Ask Oscilla: project_list, project_rename

### Return lanes (tool)

A return lane is a bus only sends feed: it has its own fader, pan, mute and inserts into the main mix, so a plug-in reverb or a tape delay can sit on one return and every lane can send to it, beside the built-in reverb and delay sends. tracks_return lists, adds, removes and sets them; a return removed takes every lane’s send to it along. The set playing rebuilds as needed, and a bounce carries the returns.

Ask Oscilla: tracks_return, tracks_send, tracks_mix, tracks_lane, tracks_clips

### Section balance

Ask Oscilla for each group's level and share of the energy per section — the lanes of the Tracks room, a project, or files, in groups the way a person hears them, over sections given in seconds, cut every so many bars at a tempo, or the whole as one. Per section per group: the mean level in dBFS with the lane gains applied and its percentage of everything there, and

which group leads. This is energy, not hearing; the ear balance is the heard version with masking.

Ask Oscilla: section_balance, ear_balance

### Show a lane's line

Ask Oscilla to show a lane's gain, pan, send or effect line in the Tracks room (or hide it) — what the A and P buttons on the strip and P's right-click menu do, so you see the line it drew.

Ask Oscilla: tracks_show_line, tracks_automate

### Show the Sounds tab

Open the Sounds tab in the Oscilla pane, beside Map and Chords, optionally with one sound's card open.

Ask Oscilla: sound_palette_show, sound_palette_set

### Start a new song

Tracks ▸ ⋯ ▸ New Song (or ask Oscilla) starts a fresh song in the room: the lanes, the Nodes patch, the knobs, the Song Map with its working-on line, and the loop are emptied together, so nothing of the last song stays on screen. Everything that was there is kept as a checkpoint first, and the line under the lanes names it; restore that checkpoint to bring it all back. When a different song replaces the lanes, the last song's Song Map and knobs are put away the same way, and the line says which song the room now holds. With a song open, the room's title is the song's name.

Ask Oscilla: new_song, checkpoint_restore, project_open

### Stems from the Tracks room

Export the lanes as stems: each lane rendered on its own — through its clips, automation, inserts, sends to the returns and its group — every stem the set's length with the tail the sends and inserts ring on for, so they line up in any DAW. Muted lanes are left out and a solo keeps only the soloed. The master chain is left out unless asked for, because a stem is what the lane gives the master. The files are numbered in the room's order, and stems.txt beside them says each one's length, integrated loudness and peak. Stems already in the folder are never replaced: a second export's files take the next free name. In the room, Export Stems… in the ⋯ menu asks for the folder — a song's Renders folder by default, else Recordings/Stems.

Ask Oscilla: tracks_stems

### Stems reinforce

Open a stem pack in Tracks, then open Reinforce. Choose Sub, kit hits, Hook or Pad and press Reinforce. New parts appear beside your stems. A rerun replaces the previous reinforcement lanes. With no MIDI clock the tempo is read from the audio, overruling the master’s number when they disagree; with no keys lane the chords are read from the bass so Pad and Hook can still follow them; the summary says where each came from and names any part it could not make.

Ask Oscilla: stems_reinforce

### Stereo width, band by band

Ask Oscilla how wide a mix is in each band — sub, bass, low-mid, mid, presence, air: the

correlation of left and right, the side against the mid in dB, the balance between the channels, the band's level, and what a mono fold-down would lose there. The lows should be narrow and the top may be wide; when the sub or bass is wide the tool warns, because a club system sums them to one speaker. A reference file can be read beside it with the differences listed. The texture fingerprint gives one width for the whole file; mixing decisions want it per band.

Ask Oscilla: stereo_bands

## The automation lines

Ask Oscilla what lines the Tracks lanes carry — gain, pan, sends and effect settings — and, while the set plays, the send and effect values the lines have put on the mixer now.

Ask Oscilla: tracks_automation

## The beat grid

Ask Oscilla for the tempo to a hundredth of a beat per minute and where beat 1 falls — the two numbers every bar-based reading needs and the plain analysis could only guess (129.2 for a track that is 130.0). The kick-band and wide onset curves are folded on every tempo near the coarse candidates in steps of a hundredth and scored on the comb of beats itself, the phase to half a millisecond; the downbeat is the beat phase that carries the most sub and kick. Beatless music is called beatless, with no tempo invented for it. The tempo in fifteen-second windows shows whether a performance drifts.

Ask Oscilla: beat_grid, timing_vs_grid, groove_map, onset_grid_image, section_map

## The ear meter: what is heard of each part

How loud each part of a mix sounds with everything else masking it — not how much energy it has. The lanes of the Tracks room (or a project, or files) are gathered into groups (groups: {name: [lane names, indices or paths]}; the unassigned become ‘other’) and measured over a window: each group’s loudness alone and in the mix (partial loudness), the percent masked, its share of what is heard, the transient (90th-percentile) partial loudness, which group masks it most and in which ERB bands. The model is an outer/middle-ear weighting, a 40-band gammatone filterbank on the ERB scale from 50 Hz to 7 kHz, compressive specific loudness and partial loudness above the rest of the mix. A part whose loudness alone does not move while its heard share falls is the part that ‘got lost’. image gives the auditory spectrogram of the mix and each group’s heard share over time.

Ask Oscilla: ear_balance, project_list, tracks_rename

## The groove, folded onto a bar

Ask Oscilla where the kick, sub, bass, clap and hats land: a section is folded onto one bar of sixteen steps per band, each step's strength from nought to one, a text row such as X . . . X . . . X . . . X . . ., the pattern in words (four-on-the-floor, on 2 and 4, off-beat eighths, sixteenths), each lane's lean from the sixteenth grid in milliseconds, and the swing. This is how a groove is read to be cloned — bass on the off-beat eighths, short and gated — without folding bars by hand. The tempo and the bar's start come from the beat grid when not given. The map also says, per step, how far the hits sit from the line in milliseconds — the groove another clip can take.

Ask Oscilla: groove_map, beat_grid

## The hand mix moves

The most important part of working two ways: the mix moves the person made by hand since the agent last looked, written as differences — choir fader −4 dB, arps pan 0 to −0.2, bass muted, a lane added. Each carries a time and a lane. The difference is what the ears decided, and the agent learns the habit from it: a choir kept four dB under its own level is kept there on the next track without being told. Oscilla reads it with a time, or everything since its last look.

Ask Oscilla: tracks_changes

## The lanes of a project

Ask Oscilla what lanes a project has: name, file, level in dB, place on the timeline, and whether it is muted. For the project open in Tracks it reads the knobs as they are in the room — level, pan, sends, effects, mute and solo — so what was dialed in by hand can be read back.

Ask Oscilla: project_list, project_tracks

## The live mix, in one call

Ask Oscilla where the mix stands right now — as the faders are, not as last saved: every lane's fader in dB, pan, mute, solo, reverb and delay sends, inserts with on or off and their settings, clips with gain and fades, automation lines, color; the master; the tempo and whether the song plays; and how many hand edits the agent has not yet been sent. One call, the whole picture.

Ask Oscilla: tracks_mix

## The melody, note by note

Ask Oscilla which note sounds when: the tune of a sung idea, of a sample before it is re-pitched, of a lead that may have missed a note. A harmonic-sum pitch tracker reads the strongest line every ten milliseconds between 80 and 1000 Hz (five harmonics, each bin's noise floor taken out first), smooths it, and keeps runs of one semitone of sixty milliseconds or more as notes, each with its cents off the name and a confidence. Polyphonic material gets the strongest line and, every tenth of a second, the three pitch classes that carry the most energy. A loop labeled D minor whose notes are E minor is caught here before a transpose puts it in the wrong key. Drums and noise come back with no notes and a note saying so.

Ask Oscilla: pitch_track

## The mixes kept

The mixes kept for this song, newest first — name, when, how many lanes. The room's ⋯ menu lists them under Load Mix.

Ask Oscilla: tracks_snapshots

## The sections, from the bands

Ask Oscilla for the structure of a track as its bands tell it, every eight bars: the sub, bass, low-mid, mid, presence and air levels of each row in dBFS and against that band's loudest row, the row's loudness, and the share of its beats that carry a kick. From those come proposed sections — intro, groove, build, drop, breakdown, low-end break (the sub full with the top filtered off), beatless low, outro — each with its bars and the numbers that argue for it. The loudness timeline finds sections only where loudness moves; a dance track's structure lives in the bands, and this reads them.

Ask Oscilla: section_map, beat_grid

## The Sounds tab

Map says when things happen, Chords says what notes, Sounds says what each part sounds like and why it was chosen. Oscilla lists every sound in the song in the Sounds tab of its pane — a hex with the role (DR drums, LO low end, CH chords, AR arps, ME melody, AI air, TX texture, FX transitions), the name, the sections it plays in — with a line on why it picked it. Click a section hex to see only the sounds in it; click a sound to open its card. Play it alone, in the mix at its bar, or what it was made from; tap the words that say what you hear (too bright, muddy, love it…), mark it keep, change or drop, press 📍 There while a clip plays to point at the exact moment, write anything. Send to Oscilla, and its answer appears under the card; a dot marks a note waiting for an answer, a tick an answered one.

Ask Oscilla: sound_palette_set, sound_palette_get, song_map_get, inbox, inbox_reply

## The Sounds tab, read

What the Sounds tab holds right now: the parts Oscilla published, the feedback marked on each card but not yet sent, and the palette's notes with their replies.

Ask Oscilla: sound_palette_get

## The takes, kept

Every take recorded in the Tracks room is kept — take 1, take 2, take 3 on the same lane, never overwritten — with where it landed in seconds and in bars at the tempo of the moment, the key read from it, the input it came in on and when. A new take is announced in the agent's inbox as it lands, with the same numbers, so the agent can go and listen, measure its timing and tuning, and build the next part around it. Ask Oscilla to list the takes, or the takes on one lane.

Ask Oscilla: tracks_takes

## The Tracks tempo

The tempo of the Tracks room sets its bars, the metronome, the grid and the phone remote's bar display. Type it in the tempo field at the top of Tracks, tap it with TAP, or let your assistant set it to match the song.

Ask Oscilla: tracks_tempo

## Timeline analysis

Ask Oscilla for a timeline of a file: second by second (or any step) its loudness, peak, brightness, stereo width — whole and per band — and the level of eight bands from sub to top, plus the onsets it hears. It shows the arc of a piece in numbers: where the energy rises, where the width collapses, whether a placed hit landed. Give from and to to look at one passage.

Ask Oscilla: timeline

## Timing against the grid

Ask Oscilla how a layer sits on the grid: every hit's distance from its nearest sixteenth in ms, the median lean (plus is late, minus early), the mean, the spread of the middle half, the ten worst hits with their bar, beat and sixteenth, and a verdict in words. The hits are energy rises, read echo-robustly — a rise within 300 ms of a hit that is under a quarter of it is taken for an echo and skipped, which is what fooled the first detector on The Empath's Path, where the flute was 200 ms late. Read the kick band for drums, everything for a melodic layer, the highs for hats; the tempo is detected when it is not given.

## Tonal lines

Ask Oscilla for the partials that will not stop: spectral peaks at least a given prominence above their neighborhood, tracked across 23 ms frames within one bin, and every line that holds for the minimum seconds listed with its frequency, start, end, length, level and prominence, longest first — the bowl's 3.95 kHz whistling for eight minutes on The Empath's Path, a resonance, a mains hum. Then a narrow notch on the equalizer card (Q up to 40) takes it out.

Ask Oscilla: tonal_lines

## Tracks

Drop audio files or a stem-pack folder or zip to make lanes — on the room, or on the Tracks item in the sidebar from the Library. Play, stop, loop and Fit control the view and transport. Each lane has gain, pan, mute, solo, sends and insert effects; the master has its own processing. Open Reinforce to add synthesized parts beside the stems. Save Tracks keeps the assembly as a .tracks file; Save as Song… makes the set a song that keeps its bounces, moments and history inside itself (Recordings, songs and what a song keeps says how the two differ). Click or drag in the time ruler to move the playback cursor: Play starts from it, and while the set is playing it carries on from where you put it. Stop pressed a second time returns the cursor to the start. Reinforce needs separate stems (a bass lane, a drums lane, a keys, guitar or strings lane); on a set without them it says so. Every lane has a level meter (after its inserts and fader), the reverb and the delay each show what they are sending back, and the master shows what goes out left and right — turn a send up and its meter moves. New filter inserts start at gentle settings: a high-pass at 90 Hz, a low-pass at 8 kHz, a wide band around 1 kHz. Click an insert to edit it: a plug-in with a window of its own opens that window, anything else opens its sliders; right-click for bypass, a plug-in's parameter list, move and remove. The bar is grouped by job: on the left the set's title, its tempo (drag it, double-click to type, or TAP it), the metronome and the count-in; in the middle the counter, then Loop, ⏮, Stop, Record and Play; on the right + Track, Add, Bounce and a ⋯ menu with Open, Save, Save as Song, Reinforce, Split, Undo and Redo. Messages, the recording time and bounce progress show in the status line under the lanes; + FX under a lane's knobs, and under the master's, adds an insert effect or a plug-in; LIM switches the master limiter. A meter turns red at full scale. The blue line marked start is where Play begins; the white line is the playhead. A project opened here brings its tempo to the delay, and from then on every change made in the room — levels, pans, sends, effects, clips, mutes — is written into the project at once; when Oscilla adds, changes or reopens the project's lanes it starts from what you have done, and the set as it was before stays one Undo away. A bounce of a performance joins the lanes already here when those are bounces too, so a song can be built part by part; an Arranger take or an opened project replaces the set. A bounce lines every lane up to the slowest plug-in’s look-ahead and drops that shared delay from the front, so nothing comes out late. Clips: every lane is a clip from the start. Drag a clip to move it (it snaps to the grid; hold ⌘ to move freely); drag its left or right edge to trim it — the pointer turns into a resize arrow there and grips show on the edges — and its top corners for fades. Trims keep the whole file, so an edge can be pulled back out. Split (⌘E) cuts the selected clip at the playhead while the set plays, at the cursor when it is stopped — or every lane, with nothing selected. Right-click a clip for Split at Playhead, Duplicate, Delete, Remove Fades, Reset Clip Gain and a note to Oscilla. Delete removes a clip, ⌘D duplicates it after itself, ← → nudge it a grid step (⇧ four), ⌘↑ ⌘↓ change its gain a decibel. Undo and Redo (⌘Z, ⇧⌘Z) cover every clip edit. Play and Bounce render the clips, so a lane placed later on the timeline is heard later. Automation lines: the A and P

buttons on a lane's strip show its gain line (dB) and pan line over the clips — click to add a point, drag a point to move it, double-click or right-click a point to remove it (the right-click menu clears the whole line too); between points the line is straight, beyond the ends it holds. Right-click P for the reverb and delay send lines and any effect line Oscilla drew. Play and Bounce follow every line: gain and pan sit on top of the strip's knobs as trims, a send or effect line takes over its knob while the set plays. Undo covers them. Under the timeline sits the zoom-and-scroll bar: its capsule shows where the view sits in the arrangement — drag its middle to pan, its edges to zoom, pinch, click outside it to jump; – and + step the zoom, Fit shows the whole arrangement. Double-click a lane’s name — in its title strip or in the badge at the top of its channel strip — to rename it, and the set’s title in the toolbar to name the set. Right-click the badge for the lane’s color: a palette, any color from the color panel, or back to automatic. The counter by the transport shows the playhead while the set runs and the cursor when it doesn’t — min:sec, SMPTE, bars|beats at the set’s tempo, frames or seconds (click to change, right-click for the frame rate); its nudge keys move the cursor a frame. With a clip selected the arrow keys nudge it a grid step (⇧ four steps); ⌥ moves it one sample, ⌥⇧ one millisecond. Right-click a lane for a note to Oscilla at the time under the mouse (or on the lane's badge, about the whole lane): the words go to Oscilla's inbox with the lane, the time and the bar, Oscilla takes a turn on them, and an amber pin stays on the lane where you clicked until it answers — hover the pin for the note and the reply. Record (the red button beside Stop) records the audio input while the set plays from the cursor onto the armed lane (R on a strip; + Track makes an empty armed one) or a new lane; the metronome and a one-bar count-in sit beside it. Drop a file onto a lane to place it there at the time under the mouse; drop it anywhere else for a new lane. The space bar plays and stops; ⏮, Home or Return goes back to the start.

Lanes: click a lane's strip to pick it, ⇧-click for a range, ⌘-click to add or drop one, ⌘A for all, Esc to let go. With lanes picked, ⌘D duplicates them — each copy goes right under its original with the same file, clips, knobs, sends, effects, automation and color — Delete removes them (Undo brings them back), M mutes or unmutes them, ← → move them together a grid step, and dragging a clip on one of them moves them all. The same actions are on the right-click menu of a lane's name. The name has a row of its own at the top of the strip; drag the strip's right edge to widen the headers. Dragging in the time ruler moves the playhead, and while the set plays the sound follows the drag. A click anywhere in the lanes away from a clip — or in the space under the last lane — puts the playhead there; a double-click plays from there. A line runs under each lane so the lanes read apart. A clip whose file is still being written shows reading… until its waveform is ready. An armed lane shows its input level live on its strip, so a signal can be seen before Record. The time ruler and the master stay at the top while the lanes scroll under them. Drag a clip up or down to move it to another lane; hold ⌥ to drop a copy instead, and ⌘ to place it off the grid. ⌘J (Consolidate, also on a lane name's menu) renders a lane's clips, fades and gain and pan lines into one file (its send and effect lines stay on the lane) in the project's Stems folder; Undo brings the clips back. ⌥-scroll changes the lanes' height, ⌘-scroll zooms time; ⇧-drag in the ruler sets a loop brace and ⌘L loops. The ruler counts bars and beats at the set's tempo — bar numbers, a tick every bar and every beat when zoomed in, and bar lines down every lane — and clips, edges, drops and the loop brace snap to that grid: a sixteenth, an eighth, a beat, a bar or a few bars, whichever the zoom shows (⌘ is free). Right-click the ruler for minutes and seconds instead. Z zooms the view to the selection — the selected clips, the loop brace, or the picked lanes — and scrolls to it; F turns Follow on and off, which pages the lanes along with the playhead while the set plays; a drag over empty lane ground is a marquee that selects every clip under it, across lanes (⌘ adds to the selection). Clips never overlap: a clip moved, copied, duplicated or trimmed over another cuts away what is under it, as in Live, so what you see is what plays. The Tracks menu also imports: 0RKESTRACKER stems with their manifest, and an Ableton Live set’s audio tracks, clips, tempo

and locators. Return lanes are buses only sends feed, each with its own fader, pan, mute and inserts; a lane’s send to one is post-fader by default or pre-fader by choice. Group lanes are folders whose members sum into a bus with its own fader, mute and inserts; solo on a group solos its members, and groups nest. While the set plays, the Song Map panel follows it: its cursor rides along the map's bars and the section playing now is outlined amber, bar 1 being the start of the set — the same mapping a double-click on a section plays from. You can make or change the map by hand too: with a set loaded, Start a map from this set, Split at cursor, drag the gap between two sections to move a boundary a bar at a time, right-click a section to rename, merge, set its energy or delete it, or draw the energy curve with the mouse — the same map Oscilla writes. When Record starts on a Bluetooth output, the status line says that what you hear is about a quarter second behind the set; play to the count-in, or use wired headphones, the Mac's speakers or an interface. Under the master's meters a loudness readout shows momentary, short-term and integrated LUFS and the true peak as the set plays; click it to start the integrated reading again.

Ask Oscilla: project_bounce, stems_reinforce, tracks_rename, tracks_record, tracks_place, tracks_lane, tracks_loop, tracks_view

## Tuning from the partials

Ask Oscilla how far a track or a note sits from A440: the strongest partials are found to a fraction of a bin, each named with its cents, and their cents are averaged on the circle (minus fifty and plus fifty are the same place) weighted by level, with a spread that says how much they agree. A record forty cents sharp on every partial reads as such, and its key can then be read with that in mind rather than split between two names. For one note the tool finds the fundamental from the harmonic series — a sub on D1 whose partials sit at 73, 110 and 147 Hz is named D1 from twenty-five hertz up, where a plain pitch reading of the fundamental goes wrong. Given several stems at once, it says which ones are out of tune against the rest, and by how much to re-pitch them.

Ask Oscilla: tuning

## Two mixes, compared

Ask Oscilla how two kept mixes differ, or a kept mix against the song now: per lane the fader, pan, send, mute and solo differences with both values and a line of words, the lanes only one side has, and a summary. This is how the agent reads what the person changed in its mix and explains or adopts it. For an A/B by ear, load each while the song plays, or bounce each.

Ask Oscilla: tracks_compare

## Two moments, compared

Ask Oscilla what differs between two moments, or between a moment and the song now — the A and B of a decision. It lists the card settings that differ with both values, the cards only one side has, the knobs that moved, the lanes whose level, pan, mute, solo or clip count differ, the lanes only one side has, and each side's latest render so both can be played. When nothing differs it says so. In the Moments window, Compare does the same against the song now and writes the summary under the list.

Ask Oscilla: checkpoint_diff, checkpoint_list

## Undo a clip edit

Ask Oscilla to undo the last clip edit in the Tracks room, or redo one.

Ask Oscilla: tracks_undo, tracks_clips

## Warp to a grid

Ask Oscilla to warp a file to a grid — markers like Live's. The beats are found (or given as pairs of seconds in the file and seconds in the result), each is paired with its line on the target grid at the tempo from the offset, and the sound between beats is stretched to fit with a pitch-preserving WSOLA — windows overlapped at half their length, each taken from the place that best continues the last, so hits stay hits — then the segments are joined with five-millisecond crossfades: no gaps, no overlaps. It was asked for when a generated track's onsets fit the grid only half the time and every layer laid on it drifted; slicing at the beats and re-laying the slices had left a seam at every one. Modes: beats for drums and anything with hits, complex for mixed material, texture for pads. A beat that would stretch its segment past the limit is taken for a false beat and dropped. A lane can be warped to a project's tempo as it is added.

Ask Oscilla: warp, project_add_track, timing_vs_grid

## What came from the phone

Settings ▸ Phone lists the last presses from your phone — what each did, whether it happened, and how long it took — and Copy diagnostics puts the last fifty on the clipboard for a bug report. On the phone, a line under the header says each press is sending, done or not done (with the reason); a press made while the connection was down is sent again when it returns, and applied only once.

Ask Oscilla: remote_log

## What was heard

Stop listening and read it: the numbers every half second, the tempo with where bar 1 falls, the key, and the sections on the bar grid — intro, groove, build, drop, breakdown, low-end break, outro — each with the band levels that argue for it. A silent input says it was silent and why.

Ask Oscilla: listen_stop

## Where a project's lanes are kept

Ask Oscilla where each lane of a project is kept: inside the package (Stems/, Renders/), near it — your own file beside the package, stored with ../ so an album folder moves whole — or absolute, the file's own place; and whether each file is there. With rewrite it turns absolute lanes that sit inside or near the package into relative ones, in place: the fix for a project whose lanes were added before paths were stored relative.

Ask Oscilla: project_paths, project_add_track, project_tracks