NUMERICAL

# Konstrukt

## Manual

Build the instrument you play.

numericalaudio.com

# Contents

**Getting started** 4
- Standalone 4
- AUv3 4
- Start from a template 5
- Build your first patch 6
- Try a factory preset 6

**Using the app** 8
- Control surface 8
- Sequencers 13
- Header & edit mode 21
- Graph editor 22
- Fragments 28
- Slide-up keyboard 29
- Transport & ports 29
- Rack 30
- MIDI Learn — hardware controllers 33
- Session restore 33
- Settings 33
- Presets 34

**Recipes** 36
- Arpeggiate held chords 36
- Scene-switch a groovebox 36
- Slide bass 36
- Sequence a chord progression 36
- Debug a silent patch 36

**Reference** 38
- Node catalog 38
- Control catalog 145
- Factory preset list 159
- Troubleshooting 160

# Konstrukt Manual

Konstrukt is a workbench for building **MIDI processors and performance surfaces by hand**, with no code. Patch a node graph, lay knobs, faders and step grids over it, and play it as a **standalone app** or an **AUv3 plugin**.

- **Standalone** — its own transport, MIDI ports and built-in instruments, so a patch sounds right away.
- **AUv3** — a MIDI processor in AUM, Logic Pro, Cubasis, Drambo or Loopy Pro, following the host's transport and tempo.

Konstrukt runs on iPad and iPhone (landscape), iOS 17 or later. On an iPhone Duo it is a landscape phone folded; opened, it lays out like an iPad and can be held either way — upright, the inspectors come up as a sheet in the lower half with the surface or graph live above. On any iPhone the status bar stays hidden, so the whole screen is the app. A Mac version is coming.

> **Konstrukt is a MIDI processor first.** To hear a patch, add instrument nodes, turn on **Built-in Sound** in Settings, or route **MIDI Out** to a synth.

---

# Getting started

## Standalone

1. Launch Konstrukt. **First Light**, a self-playing patch, loads; press **play**. Later launches reopen your last patch (Session restore).
2. The first launch shows a short welcome. It ends on **Show me around** (a tour of the real UI), **Start from a template** or **Just play**. Reopen it, or two hands-on lessons, from **…** › **Tutorials**.
3. Factory patches bring their own instruments. For your own patches, add **Analog** or **Drums** nodes, or turn on **Built-in Sound** in Settings.
4. To play another app or your gear, pick the port from the **sync** pill.




## AUv3

1. In your host, add **Konstrukt** as a MIDI processor (type `aumi`).
2. Route a MIDI source into it and a synth after it. Factory patches send each part on its own channel (drums on 10).
3. Press play in the host; tempo-synced nodes follow its tempo.

The plugin takes any view size. Below about 1000×640 the toolbar folds into compact chrome; below 480×260 the window pans over a full-size surface.

## Start from a template

Open the preset browser (the **folder** or the patch name) and tap **New Patch…**. The sheet asks **What do you want to build?** Pick one of five starting points and answer its few questions; every answer has a default, so **Create** works straight away. Where a template plays something, **Plays** picks what: **Built-in sounds** (Konstrukt's own Analog and Drums), **My plug-ins** (hosted AUv3 instruments, their windows on the surface; standalone only) or **External gear** (an External Instrument per part on the port you pick, no sound of its own). Questions that only apply to one answer — the plug-in list, the port and channel — appear as you pick it.

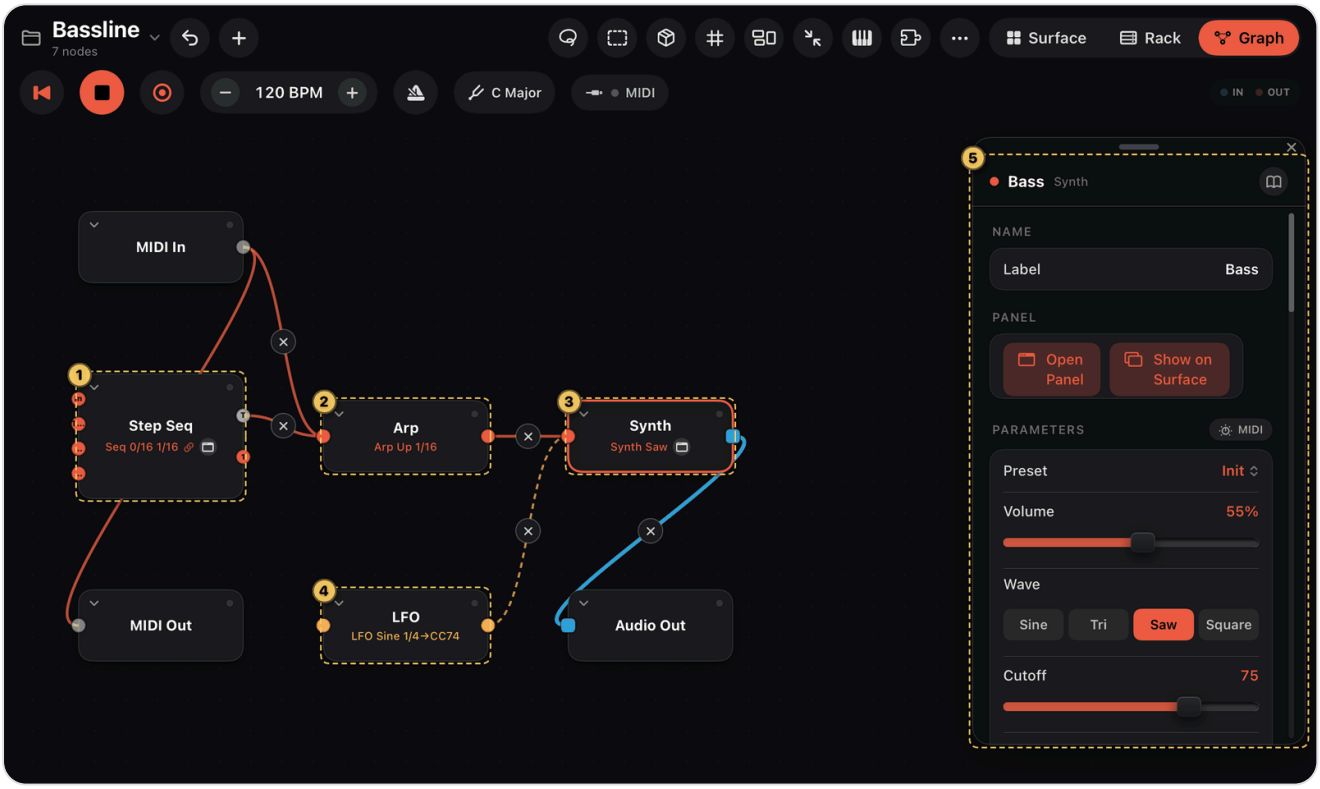
| Build | What you get |
|---|---|
| **A control surface** | Knobs, faders and pads that send MIDI, nothing else: a synth panel on the standard synth CCs, an eight-strip mixer (CC 7 under CC 10 on channels 1–8), or pads with knobs and faders on CCs 20–35. Pick the channel and output port. |
| **A keyboard controller** | A piano, scale, chord or MPE keyboard into what plays: a built-in Analog with filter and reverb knobs, one of your plug-ins with its window, or your gear with the synth panel under the keys. MPE into gear keeps every touch's own channel. |
| **A groovebox** | Drum pads, lead and chord keyboards, an XY pad on the lead's filter and space, scene pads, plus a **Sequence** page whose Drum Grid and Melody lane record what you play, a bass line and the mixer. On plug-ins a third page holds the four windows; on gear the parts leave on channels 10, 1, 2 and 3 of the port you pick. |
| **A generative patch** | A self-playing generator in your key — Random seeds, Life colony, Turing machine or Cellular rule — through a built-in Analog and a reverb, a plug-in, or your gear; an optional Euclidean kit underneath. Play a key to steer it. |
| **A MIDI processor** | Scale lock, in-key chords and an arpeggiator between MIDI In and MIDI Out, for use inside a host. |
| **Empty** | A blank patch. |

A template also sets the page the patch opens on. Save the result under your own name; the factory presets stay untouched.

## Build your first patch

The core loop is **graph, surface, play**. The **Nodes & Bindings** lesson (**…** › **Tutorials**) walks you through steps 5–7.

1. Open the preset browser, load **Harmony Keys** and press **play**.
2. Tap **Graph** in the page switcher: **MIDI In → processors → MIDI Out**.
3. Tap a node to see its parameters. Change the **Chord** type, then play.
4. Tap **+** → **Notes → Arp** and drop it on the wire after **MIDI In**.
5. On the surface, tap the **pencil** (edit mode), then **+** → **Controls → Knob**.
6. With the knob selected, tap the **sliders** icon and, under **Binding**, pick the Arp's **Rate**.
7. Leave edit mode, hold a chord and turn the knob.




## Try a factory preset

Every factory patch plays through its own instruments, and sends each part out of **MIDI Out** as well: the drums on channel 10, the other parts on channels 1, 2, 3 and so on. As an `aumi` plugin in a host, the patch can play the synths you put after it. Good first listens:

- **First Light** — the boot patch: a Life colony into an arp.

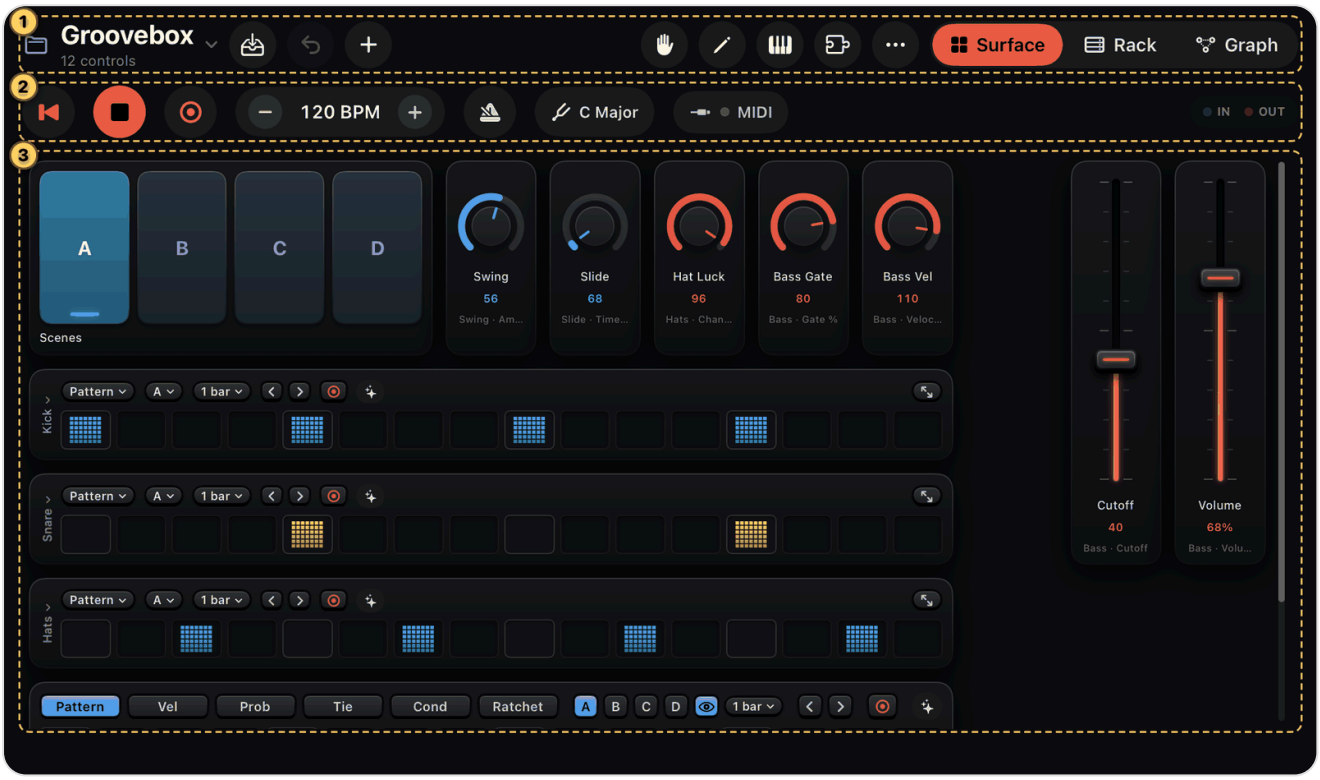
- **Witching Hour** — horror synth for Halloween: a heartbeat kick, an octave pulse, a music box that drifts against the bar, a ghost choir, bats and wind. Play the **Theremin** page — held notes glide into each other.
- **Night Drive** — synthwave: a gated-reverb kit, an octave-pumping bass, PWM strings pumped by the kick, a saw arpeggio and a lead that only comes in for the chorus. Flip the **Scenes**.
- **Echo Chamber** — dub: one drop or steppers, a round bass, the skank, a melodica deep in tape echo. Hold **Throw** to fling the skank or the kit into the delay, and play the **Siren** pad into the echo.
- **Groovebox** — drum lanes and a sliding bass, two patterns each, flipped from the **Scene Pads**.
- **Berlin School** — three Analog Seq rows: a riff, a **Roots** row that re-roots it, a filter row. Twist **Direction**.
- **Infinity Canon** — one fractal melody at three speeds at once.
- **Clockwork** — polymetric hats, clave and bass that snap back into line every few bars, or from the **Snap** pad.
- **Chord Voyage** — the Chord Seq's Chain walks three progressions, one with a borrowed E7, every chord voiced; a pad, a harp arpeggio and the roots underneath. Play your own on the **Play** page.
- **Slide Bass** — a 303-style acid line on the **Stage Seq**: Hold and Slide stages, a ratchet, three scenes over a 909, through a tube drive and a filter a sample-and-hold re-rolls every sixteenth.
- **Lorenz Skies** — hold a chord on the pads; a Lorenz attractor pushes the arpeggio around.
- **Conway Choir** — Game of Life into a wavetable choir.
- **Trance Gate** — hold a chord and draw a gate pattern on the MegaSaw.
- **Markov Mirror** — play a phrase, flip to Generate: it improvises in your style.

The rest are in the Factory preset list.

# Using the app

## Control surface

The Surface page is a grid of cards bound to the patch. A card's caption says what it drives: `Arp · Rate` is a node parameter, `CC 74` a raw CC sent on to your synth.




1. **The bar** — patch, undo, add, edit and the page switcher (Header & edit mode).
2. **The transport row** — play, record, tempo, key and sync (Transport & ports).
3. **The canvas** — the cards.

**Signal Chain** in the **…** menu adds a strip above the canvas: the patch's nodes in signal order. Tap one to inspect it beside the canvas.

Every touch goes to the card under your finger. A page taller than the window scrolls by the **scroll bar** on its edge (or by panning, with **Scroll by touch** in Settings).

A name with a ⌄ after it shows what that card plays: tap it for the route to its instrument (`Pulse → Bass`), and tap a stop to open its panel or show it in the graph. Every card that sends notes has one: lanes, grids, keyboards and pads with their own node, a scope on a generator. Knobs and faders don't, since they set a parameter. A channel strip's **Driven by** points the other way, back at the cards that play it.

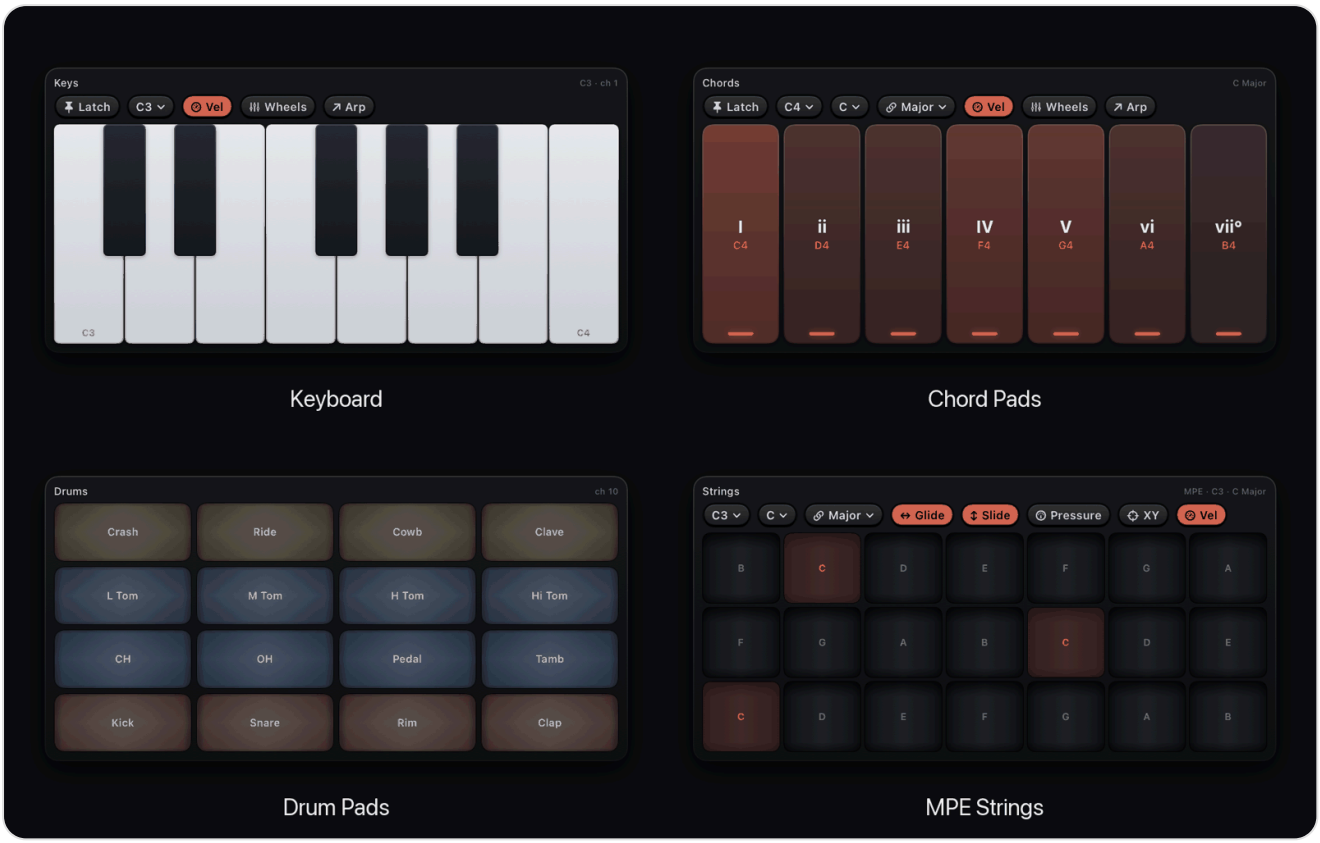
### Knobs, faders, XY pads & buttons




- **Knobs and faders** — drag to set. A fader wider than it is tall runs horizontally. **Bipolar** (inspector) fills a knob from the centre, for pan or detune.
- **XY pads** — each axis has its own binding. **Send Note** plays a note while you touch. **Record & Loop** adds a ● button: tap it, draw, lift, and the gesture loops (**Sync to Tempo** rounds it to whole beats). Each pad's **XY Pad** graph node plays a note from its MIDI output while touched, so you can wire it into a synth, and has **X**, **Y** and **Gate** outputs to wire to any parameter.
- **Buttons** — max when on, min when off; **Momentary** makes one act only while held. Bind one to a **Mute**, **A/B Switch** or **Pulse** node, or turn on **Send Note** for a trigger pad.

### Keyboards & pads

Keyboards play notes into the patch. Each is its own node in the graph, so it can run through its own processors or leave on its own channel.




- **Keyboard** — a piano; slide across the keys for a glissando.
- **Drum Pads** — a 4×4 GM kit on channel 10, or a **Custom** note per pad.
- **Chord Pads** — a pad per scale degree, triads or 7ths.
- **Scale Keyboard** — in-scale keys only.
- **MPE Seaboard / MPE Strings** — per-note bend, slide and pressure, for MPE synths and the built-in instruments.

### Keyboard toolbar

Each keyboard shows the pills that apply to it.




1. **Latch** — notes hold until tapped again.
2. **Octave** — C–1 to C7.
3. **Octaves** (Keyboard, Scale Keyboard) — how many octaves each row shows, 1 to 4; **Auto** fits the width. A tall card stacks rows above it, lowest at the bottom. The slide-up keyboard remembers its own.
4. **Root · Scale** — the key, including custom scales and **Follow Patch Key**.
5. **Slide** — finger height on a key sends CC 74.
6. **Vel** — velocity from where you strike the key.
7. **Wheels** — pitch-bend and mod wheels.
8. **Arp** — the keyboard's own arpeggiator. It keeps running while stopped, and its notes go wherever the keyboard is wired.
9. **Glide · Slide** (MPE) — horizontal bend and vertical CC 74.
10. **Pressure · XY** (MPE) — **Pressure** sends channel pressure instead of CC 74; **XY** turns each key into a small 2D pad.
11. **MPE** (Scale Keyboard) — each finger gets its own channel and the MPE pills; **Glide** bends through the scale's notes. The Arp is off while MPE is on.

**Show Toolbar** in the inspector hides the bar.

### Custom scales

Every Scale menu ends with **New Scale…** and **Edit Scales…**.

1. Choose **New Scale…** (**Copy From** starts from another scale).
2. Pick the **Root**.
3. Tap piano keys to add or remove notes.
4. Name it and tap **Done**.

A custom scale keeps its root, so “D Dorian” plays as D Dorian wherever you pick it. A patch holds up to 16; editing one updates everything that uses it.

### Patch key

One root and scale for the whole patch, shown and set by the **tuning-fork pill** in the transport row.

- New scale-aware nodes and keyboards **follow** the key, live. Followers show a **link** glyph on their graph card.
- Picking a root or scale of their own releases them; **Follow Patch Key** at the top of the Scale menu brings them back.
- **Apply Key to All** puts every lane under the key, for older patches, which load with nothing following.

### Velocity, Apple Pencil and the computer keyboard

- **Velocity** comes from the touch’s height on the key. Turn off **Vel** for a fixed **Velocity**.
- **Apple Pencil** force can set velocity, send pressure, or both (inspector). A **Pencil Curve** shapes the response.
- **Computer keyboard** — **A S D F G H J K L** are white keys from C, **W E T Y U O P** black keys, **Z / X** octave, **C / V** velocity. It plays the slide-up keyboard when open, otherwise the last keyboard you touched. Turn it off under **Settings › Hardware Keyboard**.

### Scopes & visualizers

All from **+**:

- **Monitor**, **Note Scope**, **CC Scope** — a MIDI log, a scrolling note roll and a CC trace.
- **Audio Scope** — **Wave**, **Spectrum** or **Stereo** for any channel or the master. Tap the master strip’s spectrum to float it over every page.
- **Life grids, registers and scopes** — come with their emergence nodes.




### Pages

A surface can hold several **pages**, like banks on hardware: one to play, one to mix. With more than one, page numbers sit in the middle of the bar (under the cards on a phone); in edit mode **+** adds one.

- **Move cards** with the inspector’s **Page** row.
- **Rename** or **delete** a page by long-pressing its number in edit mode. Deleting moves its cards to page 1.
- **Switch to Page** turns a button into a bank button.

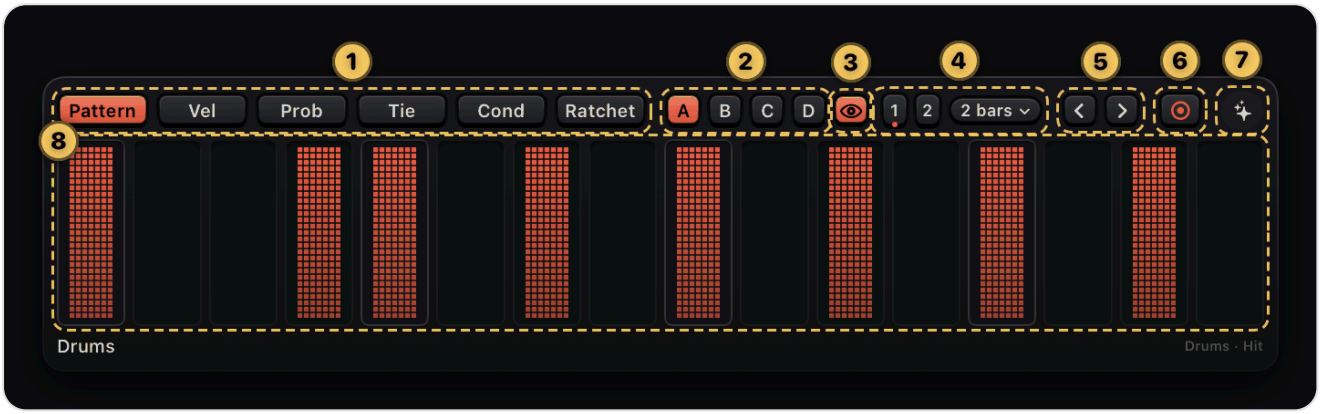
Every page keeps playing while you look at another. The Control catalog shows every card.

## Sequencers

Sequencer cards are lanes: **Drum**, **Melody**, **Chord**, **CC** and **Analog** lanes, the **Drum Grid**, the **Matrix**, the **Stage Seq** and the Piano Roll. Add one from **+ → Sequencers**; its node arrives in the graph, wired.

### Lanes

Tap a step to toggle it, drag across the row to paint, or long-press a step and drag sideways to nudge its timing.




1. **Layers** — what the row edits (Layers).
2. **Banks** — up to eight patterns, A–H (Pattern banks).
3. **Follow** — show the bank that's playing.
4. **Bars** — the clip's length, 1–4 bars, and a chip per bar.
5. **Rotate** — shift the pattern a step, every layer with it.
6. **Record** — record from a keyboard or MIDI.
7. **Player** — let a player write the bank.
8. **Steps** — the pattern. On a Melody lane each step is a pitch, snapped to the node's **Root** and **Scale**.

### Layers

The layer chips switch the same steps between six views.

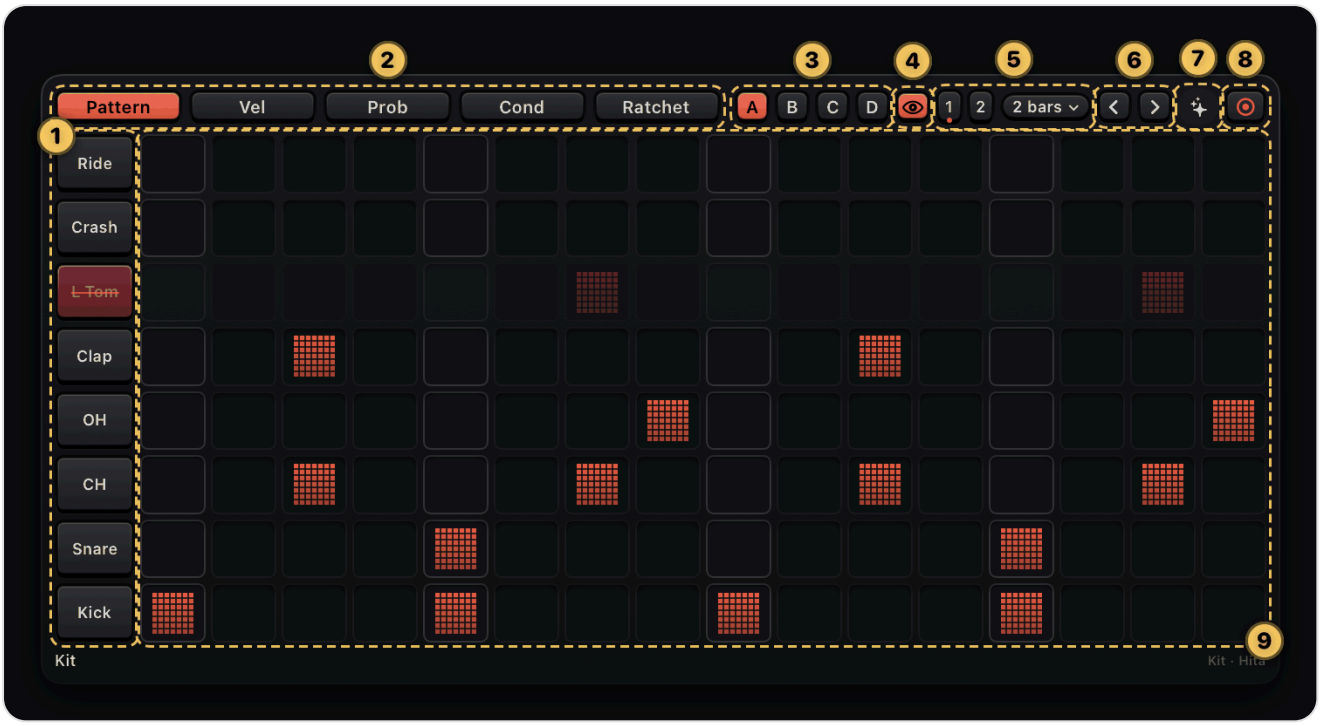



- **Pattern** — hits, pitches or CC values.
- **Vel** — velocity per step. Drag a bar; tap for full or half.
- **Prob** — the chance the step plays.
- **Tie** — hold the previous note through the step.
- **Cond** — play on one pass of several: `1:2` on the first of every two loops, `4:4` on the fourth of four.
- **Ratchet** — repeat the step ×2, ×3 or ×4, for rolls.

CC lanes have no Vel, Tie or Ratchet.

### Drum Grid

The whole kit on one card: eight rows sharing one clock, length and bank.




1. **Rows** — named after each row's note. Tap a name to mute the row.
2. **Layers** — Pattern, Vel, Prob, Cond and Ratchet, for every row.
3. **Banks** — the kit's patterns, A–H.
4. **Follow** — show the bank that's playing.
5. **Bars** — the clip's length and a chip per bar.
6. **Rotate** — every row at once.
7. **Player** — a drummer writes the bank.
8. **Record** — each hit lands on its row by note.
9. **Cells** — tap, paint, or long-press and drag to nudge.

Use separate Drum lanes for a drum that needs its own length.

### Chord lanes

A Chord lane's steps are **chord pads**: a roman numeral and a chord name, following the node's **Root** and **Scale** live.




1. **Pen** — I to VII in the lane's key, and **7th**. Pick a chord, then tap a step to place it or sweep to paint.
2. **Pads** — drag one up or down to change its chord.
3. **Tie · Voicing** — **Tie** holds the chord before; the voicing button steps through **Root**, **1st**, **2nd** and **3rd** inversions.
4. **Strum** — roll each chord over part of its step, up, down or random.
5. **Rhythm** — a second grid that decides when the chords hit. Left empty, every chord step plays itself.
6. **Rate · Length** — the rhythm grid's own, so the chords can change once a bar while stabs land on 2 and 4.
7. **Rhythm grid** — tap for a hit, drag for its accent.

An **Analog** lane is a row of eight knobs; its **oct** chip sets the range, one to three octaves above the root.

### Pattern banks

Lanes, the Drum Grid, the Matrix, the Stage Seq and Piano Rolls hold up to **eight banks**, A–H, each a complete clip with all its layers.

- **Tap a bank** to edit it and switch the lane to it. The switch lands on the next bar, so it always hits the downbeat.
- **Follow** (the eye) shows the playing bank. Tapping a bank pins it; tap the eye to follow again.
- **Longer clips** — pick 1 to 4 bars from the **bars** menu. The row shows one bar at a time and follows the playhead; tap a bar chip to pin it.

Three node parameters drive banks, and you can bind or modulate them:

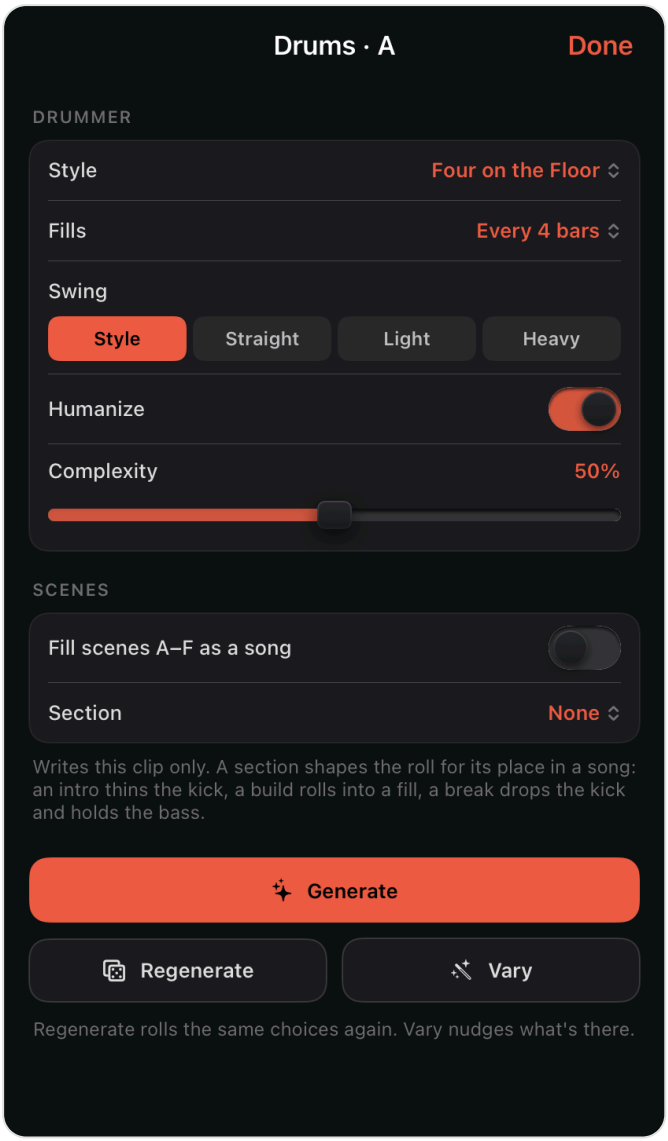
- **Pattern** — the active bank.
- **Chain** — cycle **A·B**, **A·B·C** … **A–H**, one bank per loop.
- **Launch** — **Stop** silences the lane from its next bar; **Play** resumes. A stopped lane says **Stopped**; tap it to start.

**Scene Pads** (**+ → Performance → Scene Pads**) switch every lane at once, each on its next bar, so the whole kit changes together.

### Players

The ✦ chip writes the bank you're looking at:

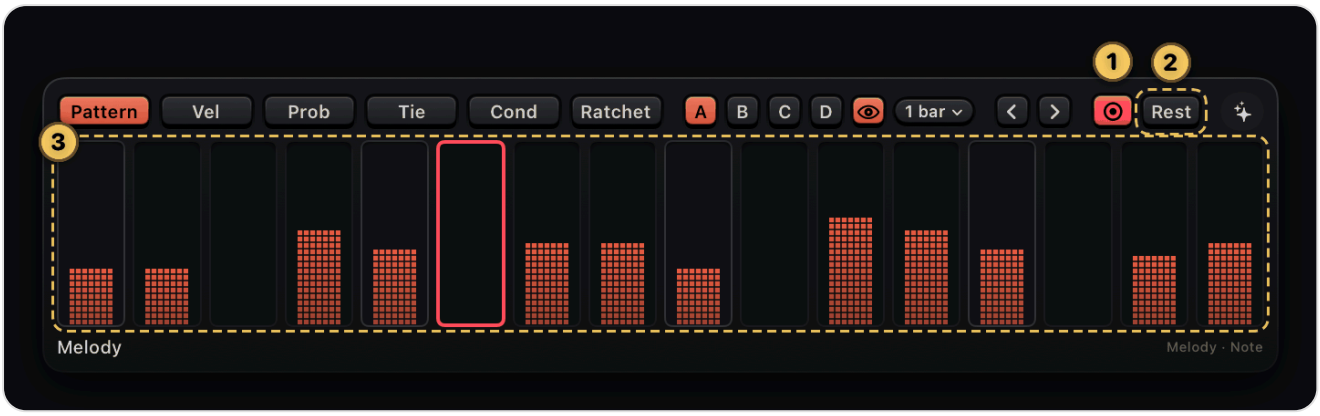
- **Drums** — a drummer, in seventeen styles, with fills and swing.
- **Melody and Analog lanes** — a bass line in the patch key, under the Chord lane's chords when there is one.
- **Chord lanes** — a progression, with rhythm, sevenths and voicing.




**Generate** writes, **Regenerate** rolls again and **Vary** nudges what's there. The result is an ordinary bank you can edit, and each write is one undo step.

### Recording into a lane

Drum and Melody lanes and the Drum Grid record notes. Arm one from ●.



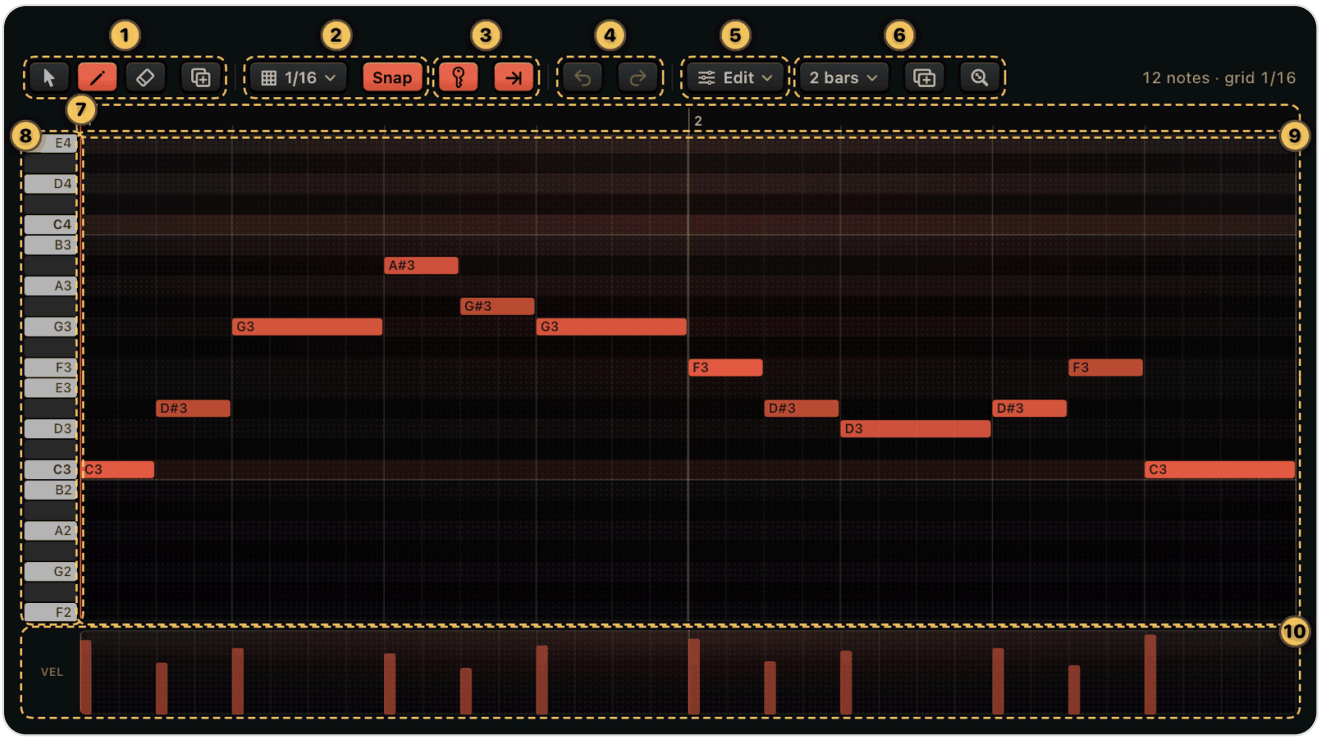

1. ● — **Step Record** or **Live Record**; the menu also has **Replace** and **Tighten Timing**.
2. **Rest** — skip a step (Step Record).
3. **Cursor** — the ringed step. Each note lands there and the cursor moves on; tap a step to move it.

**Live Record** records while the transport runs. Notes snap to the nearest step and keep their feel as a nudge; takes overdub unless **Replace** is on.

A lane records whatever is wired into its **In**, or, with nothing wired, MIDI In and the surface keyboards. A Drum lane takes only its own note, so arm every lane of a kit and play the pads.

### Piano Roll

A **Piano Roll** records what you play into eight banks, A–H. Its card has the same bank chips as a lane and a ⤢ button that opens the editor. Bank switches land on the next bar, and **Scene Pads** switch piano rolls with the lanes.




1. **Select · Draw · Erase**, then **+** to add to a selection.
2. **Grid** and **Snap**.
3. **Key** shades notes outside the patch key; **Follow** keeps the playhead in view.
4. **Undo · Redo** for the editor.
5. **Edit** — Quantize, Legato, transpose, Duplicate, Reverse, Copy, Paste and Delete.
6. **Length** — the loop's bars; **×2** doubles it with its contents.
7. **Ruler** — drag to scroll or zoom; tap to set where Paste lands.
8. **Keys** — press one to hear it.
9. **Notes** — what a touch does depends on the tool.
10. **Lane** — Velocity (drag a bar, or sweep to paint), Bend, Pressure or a CC.

Pinch to zoom, drag with two fingers to scroll. **Send to lane** on the card copies a take onto a lane's steps.

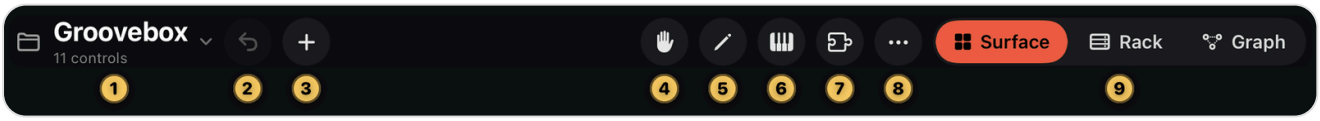
**Bend, pressure and CC.** What you played with the notes shows on them. A note's **pitch bend** rides its bar as a line, in real semitones, so a slide from C to E ends on E's row. Its **pressure** makes the bar brighter and thicker where you pressed harder. An MPE controller gives each note its own; a plain keyboard's bend and pressure are shared by every note sounding. The lane under the roll switches between **Velocity**, **Bend**, **Pressure** and each **CC** the take holds — tap its name. Sustain draws as blocks where the pedal is down.




- **Bend Range** on the toolbar sets how far a full bend reaches, for drawing it. **Auto** is what the built-in instruments play: ±2 semitones on channel 1, ±48 on MPE channels. Set it to match a plug-in or a synth.
- **Editing keeps it with its notes.** Moving, transposing, quantizing, reversing, duplicating, copying and pasting a note takes its own bend, pressure and MPE slide (CC 74) along. Deleting an MPE note deletes them too. A bend shared by a chord moves only when the whole chord does. CCs and pressure shared by every note — a mod wheel sweep — belong to the take, not a note: they stay where you played them.

## Header & edit mode

Every page's bar starts with the patch name and undo, and ends with the keyboard, plugin windows, **…** and the page switcher. The tools in between belong to the page. On the surface:




1. **Patch name** — opens the preset browser.
2. **Undo** — hold for **Redo** (⌘⇧Z).
3. **+** — add a card.
4. **Panic** — all notes off (⌘.).
5. **Edit** — move, resize and inspect cards.
6. **Keyboard** — the slide-up keyboard.
7. **Plugin windows** — open or tile floating windows.
8. **…** — the page's tools, then **Manual**, **Tutorials**, **What's New** and Settings.
9. **Surface · Rack · Graph** — the page switcher.

**Panic** releases every held note — keyboards, latches, arps, sequencers, instruments — and sends All Notes Off, All Sound Off and sustain off on all 16 channels. The transport keeps running.

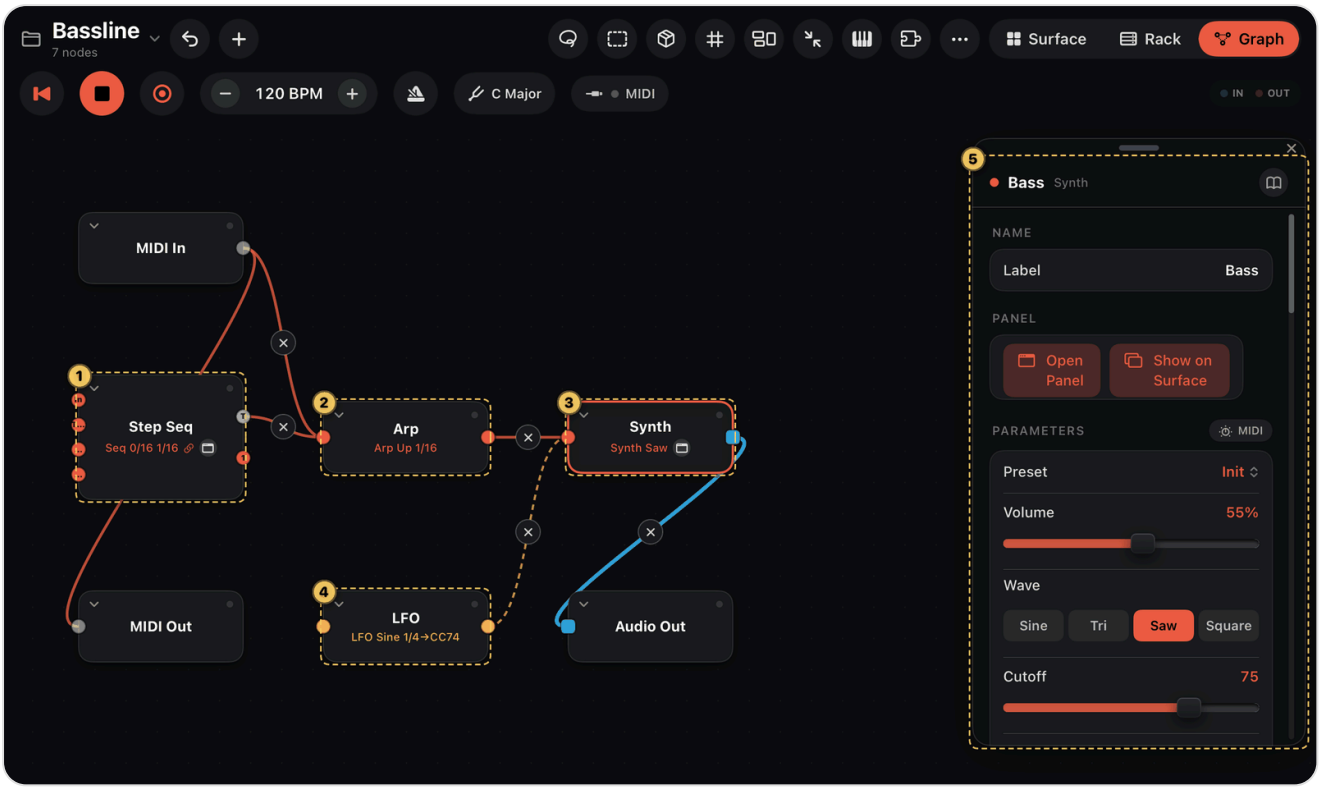
**Edit mode** (the pencil, or double-tap a card): drag a card to move it, drag its corner to resize; the others make room. The inspector (**sliders**) renames, rebinds and recolours it. A card that shows a node, such as a lane, also carries that node's settings and its **Connections**; a keyboard or pad with its own graph node shows that node's Connections, so you can rewire it without leaving the surface.

> **Tip:** long-press a card's edge to open its inspector without entering edit mode.

**Groups.** Cards that belong together — a filter's cutoff, resonance and drive, say — can share one panel. In edit mode select two or more cards (drag a marquee over them; when the page has no empty space to start one, turn on the **lasso** in the bar and drag anywhere, over cards too — it stays on until you tap it again) and tap the bar's **Group** key, or **Group** in the inspector (**⌘G**). The same bar key reads **Ungroup** while a group is selected. The cards sit in one card with its name along the bottom (up the left edge when the group is one row tall, like a sequencer lane); drag the panel or its name to move them together, and the layout keeps them together when another card lands on them. Drop one of its cards on another and the two swap places; the group keeps its shape. Tap the panel and the inspector shows the group itself: its name and colour, **Ungroup**, and its cards — tap one to edit it, or **–** to take it off the panel. While the group is selected its panel has one corner handle in place of its cards': drag it and the whole group resizes like a card, every card inside growing or shrinking in proportion and snapping to the grid when you let go — two lanes stacked in a group stay two equal lanes. It won't go below what its cards need (a lane keeps its eight columns). A card's own inspector keeps a **Group** section with the same name, colour, Select, Remove and Ungroup. The cards stay where they are when a group dissolves. A fragment's cards share a panel the same way, named after the fragment.

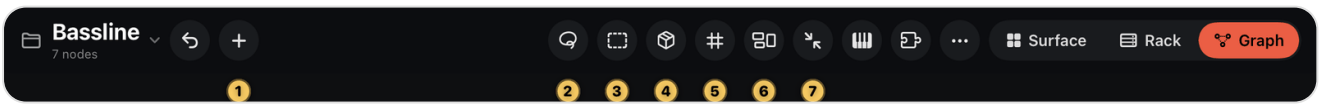
## Graph editor

The **Graph** page is the patch itself: every node and wire.




1. **A sequencer lane.** Its notes leave on a **solid orange wire** — MIDI.
2. **An Arp**, a MIDI effect wired in front of the instrument.
3. **The Analog**, selected. Its sound leaves on a **blue wire** — audio.
4. **An LFO** on a **dashed wire** — modulation, here into the cutoff.
5. **The inspector** — the selected node's parameters; the **book** opens its entry in this manual.

**Any output can modulate.** An LFO, a CC Seq, aftertouch: drag it onto a node's body and pick **Modulate *param***.

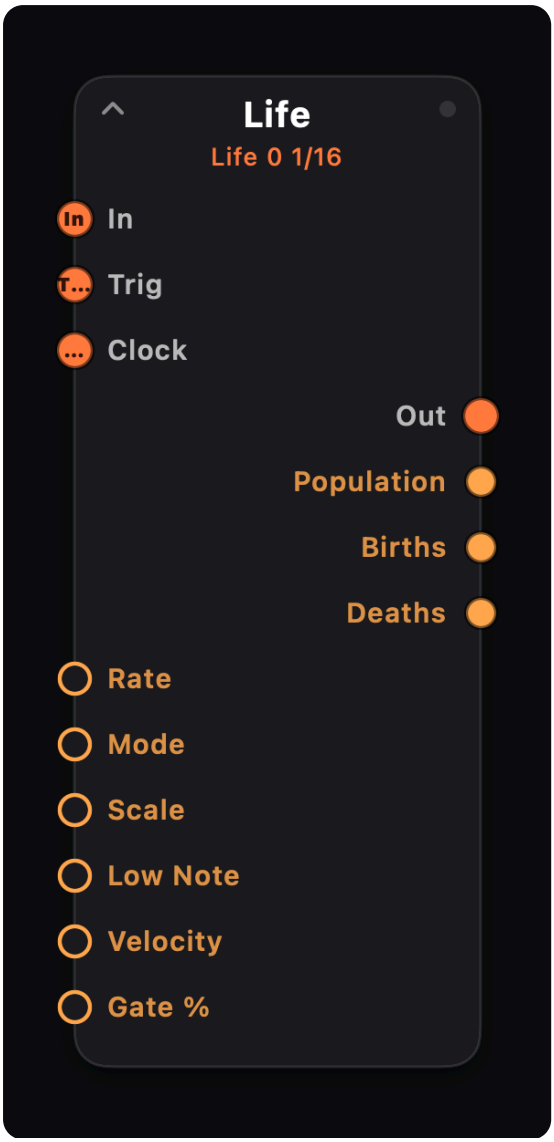



1. **+** — add a node.
2. **Lasso** — select several nodes; drag one to move them all.
3. **Frame** — a labelled backdrop round the selection.
4. **Fragment** — turn the selection into one node (Fragments).
5. **Grid** — snap and align.
6. **Tidy** — arrange the nodes automatically.
7. **Fit** — frame the whole patch. Pinch to zoom.

### Wiring basics

| Action | How |
|---|---|
| Connect | Drag from an output onto another node's **left edge** |
| Modulate | Drag an output onto a node's **body**, pick **Modulate *param*** |
| Insert mid-wire | Drag a node onto a wire, or drop a wire on empty canvas |
| Disconnect | Tap **×** on the wire |
| Wire without dragging | Inspector → **Connections** → **Add Connection** |
| Scale a mod wire | **Connections** → the wire's **range** pill → **From / To** |
| Bypass | Inspector → **Bypass** (MIDI passes straight through) |
| Colour a path | Inspector → **Color Tag** (tints everything downstream) |

Each node has an **activity LED**; the first dark one is where MIDI stops. Tap a node's **chevron** to expand its patchbay: a row per output, mod source and parameter input, so you can drop a wire on the exact parameter.




### Port colors

| Style | Examples | Role |
|---|---|---|
| Orange · solid | In, Out, Trig, Clock | MIDI in/out and external step clocks |
| Amber · filled | Population, Births, Deaths | Modulation *outputs* — drag onto any parameter |
| Amber · hollow | Rate, Mode, Scale… | Parameter *inputs* — receive mod wires |

Life node expanded — compact patchbay with clock ports, scalar parameters, and three mod outputs. Sequencer step rows stay on the surface grid, not in the patchbay.

### Scaling a modulation wire

A mod wire sweeps its target's whole range. To use a slice, set the wire's **From** and **To** (its **range** pill), in the parameter's own units. A stepped source spreads its steps across that slice: a **Counter** with `count` 2 on a wire limited to *Rows … Columns* flips a Matrix between just those two modes.

### Clocks, resets and EOC

- **Trig** and **Clock** inputs on the clock-driven generators replace the internal clock: one step per incoming note (even while stopped) or per pulse from a **Clock** node. One Clock into several generators sets their tempo in one place.
- **Reset** on the pattern sequencers restarts them at step 1.
- **EOC** (end of cycle) pulses each time a pattern wraps. Wire it into a **Trig** to step a longer sequence once per bar, or into the other lanes' Resets to keep polymetric lanes in line.

### MIDI as a mod source

**MIDI In** offers **Velocity**, **Note**, **Gate**, **Mod Wheel**, **Pitch Bend** and **Pressure** as mod sources. A node's **Out** used as a mod source follows the pitch of the notes it emits.

### Built-in instruments

**Analog**, **Drums**, **Sampler** and **Slicer** turn MIDI into audio (**+ → Instruments**). Several instruments mix together at the output.

- **Analog** — a polyphonic subtractive synth. Pick a **Preset**, then shape its oscillators, sub and noise, filter, envelopes and LFO. **Poly**, **Mono** or **Legato**, with its own **Glide**. Plays MPE with per-note bend, slide and pressure.
- **MPE** — the Analog, MegaSaw, Wavetable and Sampler each have an **MPE** section: **Pressure →** and **Slide →** send a note's pressure and slide to Level, Cutoff, Reso or Vibrato (plus Drive and Osc Mix on the Analog, Position on the Wavetable, Detune on the MegaSaw), each with a signed **Amount**. Each note moves only its own voice.
- **Drums** — an analog-style GM kit. **Preset** (808, 909, Dusty, Techno…) sets each voice; **Kit**, **Tone**, **Snap** and **Drive** shape the whole kit.
- **Sampler** — plays a sample chromatically. Load your own (its root note is detected), trim it on the waveform, **Loop** it for pads.
- **Slicer** — cuts a drum loop into up to 16 slices, one per note from 36, or loads a sampled **drum kit** that plays like the Drums node.

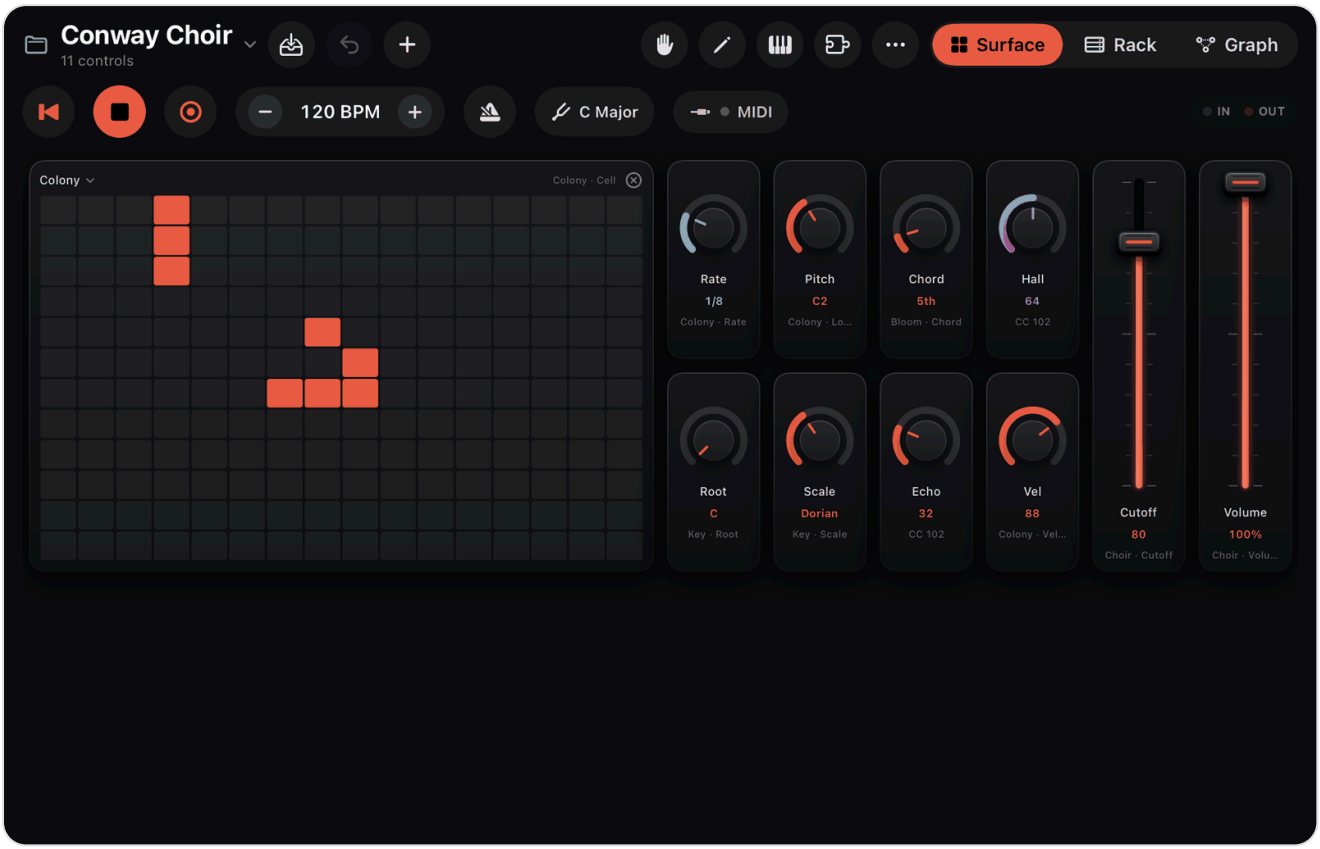



Adding an instrument turns **Built-in Sound** off, so nothing doubles.

**Your own sounds: Save Preset…** in an instrument's sound menu keeps the sound in **My Sounds**, under the factory presets. **Make Default** puts it on every new instrument of that kind. My Sounds syncs over iCloud and shares as a `.midirackinstrument` file. A patch keeps its sounds even where the preset isn't.

### Emergence nodes

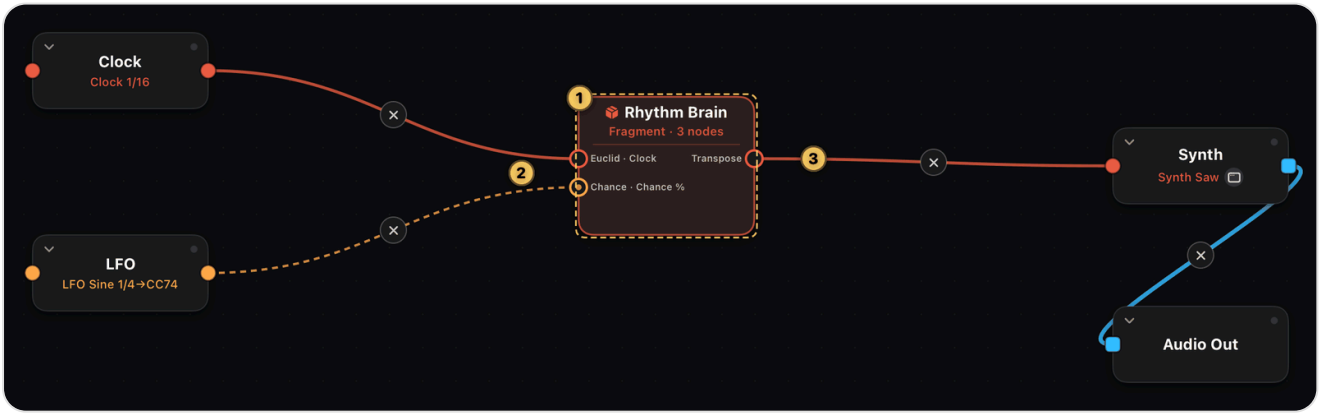
**Game of Life, Rule, Turing, Markov, Bounce** and **Attractor** make evolving patterns, each with a visualizer on the surface. See the Node catalog.




## Fragments

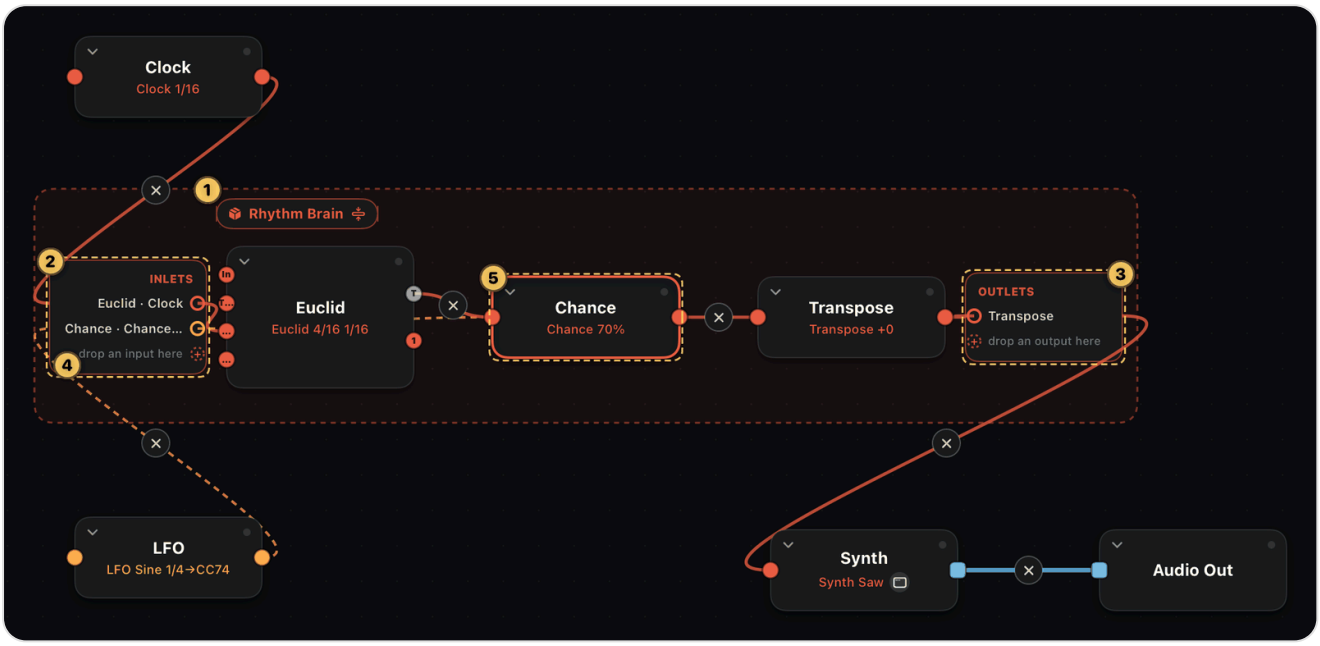
A group of nodes that works together, such as a rhythm generator or a whole voice with its knobs, can become **one reusable node**.

To make one, lasso the nodes, tap the **box** and name it. The nodes, and the cards bound to them, collapse into a single card, which is also saved to your library.



1. **The card** — double-tap it to open it up.
2. **Inlets** — the wires that came in. A modulation inlet lets an outside LFO reach a parameter inside.
3. **Outlet** — the wire that went out.

Wire a fragment like any other node; the nodes inside run as before.




1. **Name** — tap to collapse. Drag the backdrop to move the whole group.
2. **Inlets** — the fragment's inputs.
3. **Outlets** — its outputs.
4. **Empty slot** — drag an inner port here to add it, or tap to pick one.
5. **Inner nodes** — edit and rewire them as usual.

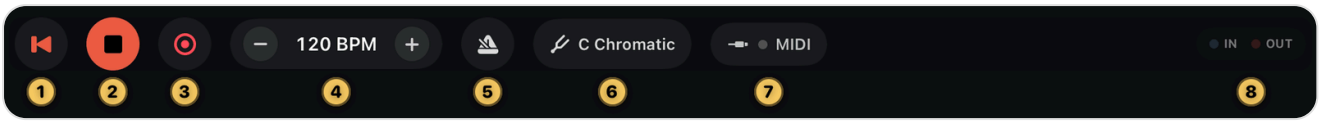
**+ → Fragments** adds a saved fragment, with its cards sharing one panel on the surface, named after the fragment, that moves as one. **Manage Fragments…** shares one as a `.midirackfragment` file. A patch keeps its own copy, so editing the library never changes old patches.

## Slide-up keyboard

The **keyboard** button slides up a keyboard whose notes enter the graph like MIDI In. Its menu switches between piano keys, scale keyboard, chord pads, drum pads and the MPE keyboards, each with its usual toolbar. Your choice is remembered across patches.

## Transport & ports

The transport row sits under every page's bar. In the AUv3 the host owns the transport.




1. **Reset** — everything back to bar 1.
2. **Play / Stop**.
3. **Record** — a take of the master and the armed channels.
4. **Tempo** — **–** / **+** by 5 BPM; tap the number in time to tap the tempo.
5. **Metronome** — hold for level, accent and count-in.
6. **Patch key** — the root and scale lanes and keyboards follow.
7. **Sync** — MIDI ports, Ableton Link and MIDI clock.
8. **MIDI activity** — blue in, orange out. A **CPU** pill appears when the load is high.

- **Stop** silences everything at once, but generators keep what they've learned, such as a Life colony.
- **Reset** (⌘↩; on a phone, long-press Play) restarts every lane, piano roll and arp from bar 1, playing or stopped. Long-press it for **Hard Reset**, which also clears what generators have learned.
- **Record** while stopped starts the transport after any count-in, so the file starts on the downbeat. The metronome is never recorded.
- **Swing** — on the Rack page the tempo pill shows **Swing** under the BPM: the patch's groove, **Off** to **100 %**, moving the odd 16ths later on every lane whose own **Swing** is **Session** (the default). A lane's Swing setting can be **Off** or an amount of its own instead; the Stage Seq's is **Session**, **Off** or 51–75 %. It is saved with the patch, and the AUv3 plays the patch's swing.
- **Chord** — once the patch has a Chord Seq, a pill beside the key shows the chord it is on and the one it moves to next, so you can play along or record a take without counting bars. Its menu lists the lane's progression with the sounding chord marked and can switch the row to Roman numerals. The AUv3 shows it in the bar beside the key.
- **MIDI Ports** — **MIDIRack Out** is always available. **Routing** sends each input **Thru** to a hardware output or straight into a hosted instrument.
- **MIDI clock** — with **Receive MIDI clock** on in Settings, the transport follows an external clock.

## Rack

The **Rack** page lays the patch out as channels: a row per instrument, Audio In or External Instrument, then the send returns and the master. A send is an audio wire with a level, an insert an effect in series, so the Graph page always shows the same routing.




1. **Identity** — tap the tile or **Panel** to open the instrument.
2. **Input** — what plays the channel; the chip under it opens its lane.
3. **Inserts** — effects in series. **+ Insert** adds one.
4. **Sends** — drag a chip sideways for its level; double-tap to turn it off.
5. **Pan and fader** — with the channel's meter.
6. **M · S · ● · …** — mute, solo, record arm and the row's menu.
7. **External Instrument** — gear on a MIDI output; its fader and pan send CC 7 and CC 10.
8. **Returns** — a row per send bus.
9. **Add** — **Instrument** (built-in or plugin), **External Instrument**, **Audio In** or **Send Bus** (the effects by family).
10. **Master** — master inserts, spectrum, meters, **Clear Solo** and **Scenes**.

## What plays a channel

To give a channel a new part:

1. Tap the channel's input capsule.

2. Choose **New Lane**, then a lane, a Drum Grid, a Matrix or a **Piano Roll**.
3. Tap the lane chip under the capsule to open it.

The capsule's menu also turns any other MIDI source in the patch on or off for the channel. While a row is **armed** (●), the keyboard and MIDI In play only armed rows. MIDI effects such as an Arp are wired in front of the instrument on the Graph page.

### Inserts, sends and the master

- **Inserts** — tap a chip to bypass it, hold it for its menu: **Replace** swaps the effect in place (sends and wiring stay, settings start fresh), **Move Up** and **Move Down** reorder the chain.
- **Sends** are post-fader; **Pre-fader Sends** in the row's menu taps one before the fader.
- **Sidechain** — the Compressor shows **Sidechain from** on its chip.
- **Mono** in the row's menu sums a stereo channel, so pan is a true pan.
- **Master inserts** process the whole mix.
- **Output** in the row's menu routes a channel to another pair of outputs on a multi-output interface.

### Mute, solo, groups and scenes

- **Mute groups** silence a set of rows with one pill under the master. Add a row from its menu › **Mute Groups**.
- **S** on a return solos the bus.
- **Scenes** on the master store the console (levels, pans, sends, mutes) and recall it, with a fade of up to 4 s.

Mute and solo come back with the session but are never saved in a preset.

### Channel settings and control

- **Properties…** (row menu) holds the node's name, colour and settings.
- **Replace Instrument** swaps the sound and keeps lanes, inserts, sends and levels.
- **MIDI Learn** in the row's menu maps a controller to the volume, pan, mutes or sends.
- A surface knob or fader can bind to any channel's volume, pan, mute, solo or send. **+ → Performance → Mixer** puts a mixer on the surface.

### Plugin windows and panels

- **Windows** float over every page; drag, resize and **tile** them. They reopen where you left them.
- **Built-in panels** — the built-in instruments and effects have a hardware-style panel, opened like a plugin window.
- **Plugin View** cards (**+ → Performance → Plugin View**) put a plugin's interface in the layout.
- A plugin that keeps overloading is muted; its inspector shows **Resume**.

## MIDI Learn — hardware controllers

Any knob, fader, XY axis, selector or button can follow a hardware control:

1. Open the card's inspector (**sliders**).
2. Under **Binding**, tap **Learn**.
3. Move a knob or fader on the controller.

**Relearn** replaces the mapping and **Clear** removes it. A button can learn a pad or key instead. Mappings save with the patch, in the app and the AUv3. Each mapping has options for **Relative Encoder**, **14-bit**, **Soft Takeover** (on by default) and **Pass Through**, which also sends the message on into the patch.

To use a controller and a keyboard together, choose **All Inputs** as the input: learned messages go to their cards, everything else plays.

## Session restore

The standalone app autosaves your patch as you work and reopens it on the next launch. **Save** still overwrites the preset it came from.

After a crash the app opens **First Light** and asks whether to restore your last patch, so a patch that fails to load can't trap you.

**On launch** in Settings can open a **Default Patch** instead (touch and hold a preset › **Set as Default**) or an **Empty Patch**.

## Settings

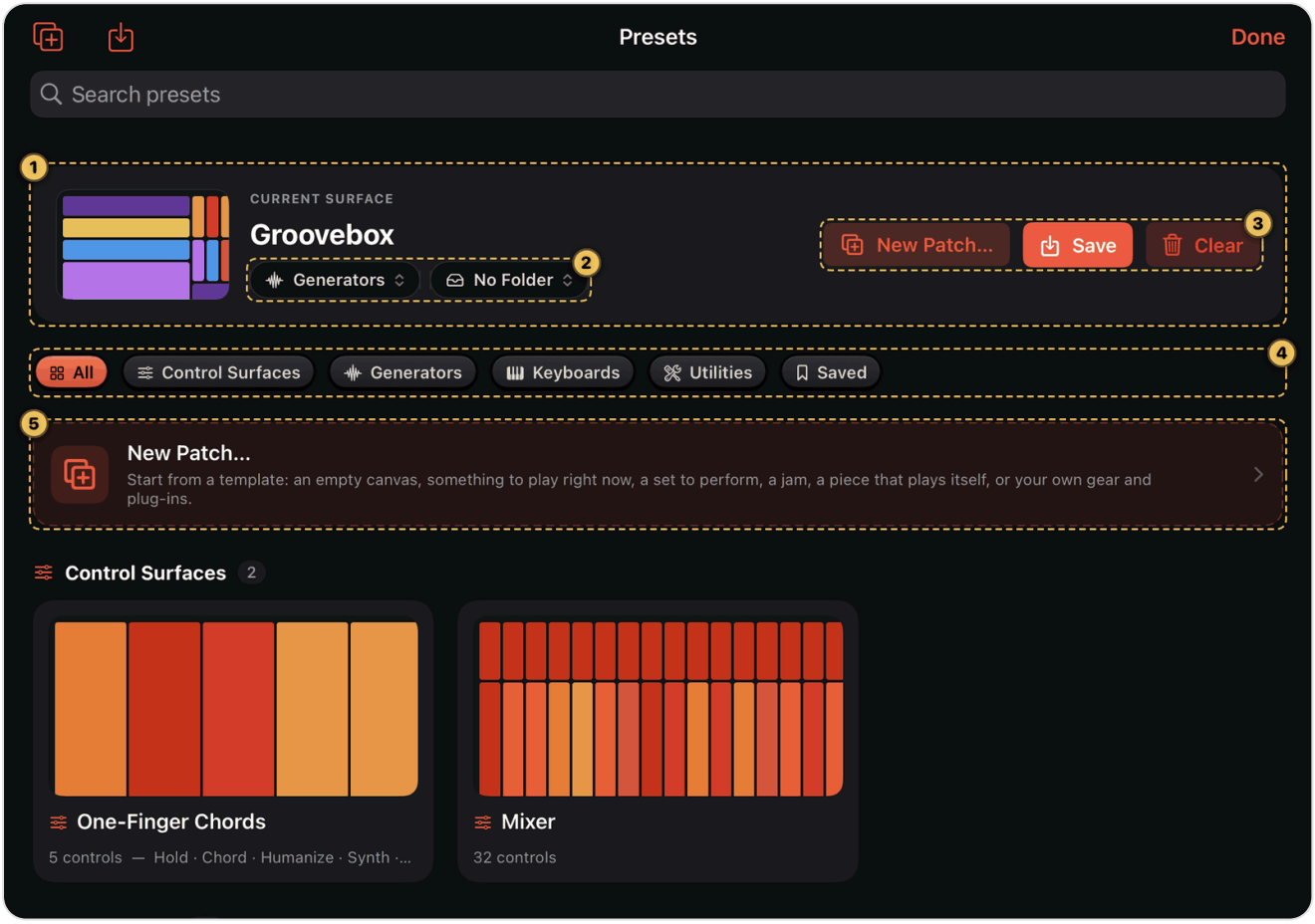
**…** › **Settings**:

- **Start playing on launch** — on by default.
- **On launch** — last patch, default patch or empty patch.
- **Audio** — buffer size and sample rate, with the load and latency they give. Raise the buffer if **Overloads** climb.
- **Plugin delay compensation** — keeps channels aligned around plugins with latency.
- **Multicore engine** — on **Auto**, tracks render side by side on your device's other cores when that pays: two or more hosted plugins, or four or more tracks. **Off** keeps everything on one audio thread. The plugin inside another app always uses one.
- **Built-in Sound** — an Analog or Drums voice for everything that leaves **MIDI Out**, for patches without instruments.
- **Sync** — send or receive MIDI clock.

- **Sync presets with iCloud** — your patches and fragments on every device, and in **iCloud Drive › Konstrukt**.
- **Hardware Keyboard** and **Haptics**.

## Presets

The **folder** or the patch name opens the preset browser.




1. **The current patch** — edit its name to save under a new one.
2. **Category · Folder** — where **Save** files it.
3. **New Patch… · Save · Clear**.
4. **Filters** — a category, **Saved**, or one of your folders.
5. **New Patch…** — the templates.

- **Folders** — long-press a patch › **Move to Folder**; long-press a folder chip to rename or delete it (its patches stay).
- **Sharing** — long-press › **Share…** exports a `.midirackpatch` file; open one to import it.
- **iCloud** — saved patches appear on your other devices and in **iCloud Drive › Konstrukt › Presets**.

In the AUv3, patches save inside the host project.

# Recipes

## Arpeggiate held chords

1. Open **Harmony Keys**.
2. On the Graph page, drop an **Arp** on the wire after **MIDI In**.
3. Set its **Mode**, **Rate** and **Octaves**.
4. Bind a surface knob to **Rate**, then hold a chord.

## Scene-switch a groovebox

1. Add a few **Drum Lanes** and a **Melody Lane**, and draw bank **A** on each.
2. Tap each lane's **B** chip and draw a variation.
3. Add **Scene Pads**. Tap **B**: every lane switches on the next bar.
4. For a hands-free arrangement, set a lane's **Chain** to **A·B**.

## Slide bass

1. Wire a bass line (a **Step Seq** or a keyboard) through **Notes → Glide**.
2. For a sequence, set Glide's **Mode** to **Always**: every note slides.
3. For playing, put a **Mono** before it and keep **Legato**: only overlapping notes slide.

Hear both in **Solo Lead** and **Slide Bass**.

## Sequence a chord progression

1. **+ → Sequencers → Chord Lane** — a I–vi–IV–V on four pads.
2. Pick a chord from the pen and tap steps to place it.
3. Bank a second progression on **B** and **Chain** them.
4. For reggae stabs, open the **Rhythm** layer and tap steps 2 and 4 at **1/4**: the chords now hit on the two and the four.

## Debug a silent patch

1. Is the **OUT** light flashing? Then MIDI is leaving.
2. On the Graph page, follow the activity LEDs; the first dark node is where it stops.
3. Splice a **Monitor** there and read its log.
4. Generators only run while the transport plays.
5. Standalone: add an instrument, or turn on **Built-in Sound** in Settings.

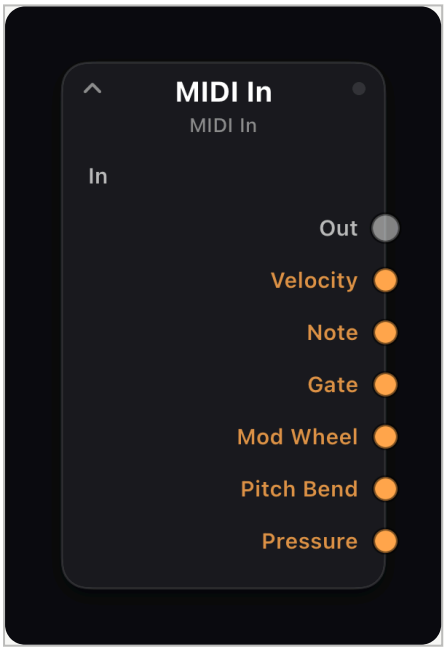
# Reference

Lookup sections — skim Getting started and Recipes first if you're new.

## Node catalog

Each entry shows the **expanded node card** as it appears in the graph editor, with ports, behavior, parameters and (when available) modulation outputs.

- **In / Out** — orange MIDI wires. Most nodes have one input and one output. Multi-port nodes label their connectors (**A**, **B**, **Trig**, **T**, **Low**, **High**, **1…n**).
- **Mod out** — amber modulation sources (MIDI In, Life, Attractor, LFO…); drag onto any parameter knob.
- **Instruments** — **Analog**, **Drums**, **Sampler** and **Slicer** are audio sinks (no MIDI out); they render sound inside the plugin or standalone app.
- **Clockable generators** — **In** (pass-through), **Trig** (step on note-on), **Clock** (step on pulse from a **Clock** node). Trig and Clock are mutually exclusive.
- **EOC** — fixed-length cyclic nodes (Drum/Step/Chord/CC Seq, **Matrix**, Euclid, Fibonacci, Turing, Rule, Gate, Piano Roll) expose an **end-of-cycle** trigger beside **Out**. It pulses once per pattern loop — wire it to another node's **Trig** / **Clock** to chain or reset patterns.
- **Rate / Grid / Time** — clock divisions from `4/1` down to `1/64`, each with a dotted (`1/8D` = ×1.5) and a triplet (`1/8T` = ×⅔) variant, synced to host transport unless a node is in Free mode.



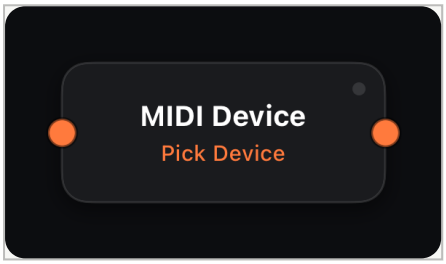
### MIDI In

`In: — (graph entry) · Out: 1× MIDI`

`Mod out: Velocity · Note · Gate · Mod Wheel · Pitch Bend · Pressure`

The graph's **entry point**. Every note, CC, pitch bend, aftertouch and program change from the host track, hardware controller, virtual **MIDIRack In** port, or the slide-up keyboard arrives here. Exactly one MIDI In node is allowed per patch.

Expand the node to expose six **performance modulation sources**. Drag **Velocity** onto Echo **Decay** so harder playing lengthens repeats, or **Note** onto filter cutoff for keytracking. **Gate** is high while any key is held.




### MIDI Device

`In: 1× MIDI (merge) · Out: 1× MIDI`

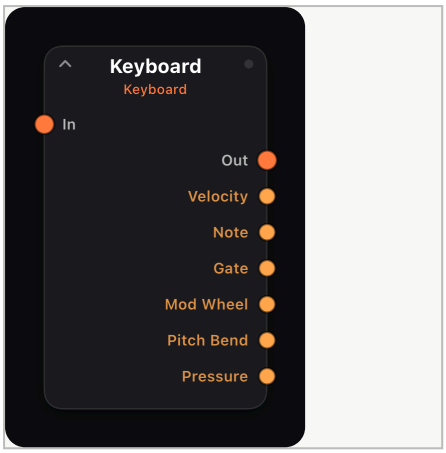
**Standalone only**: taps the MIDI of **one hardware input device**, picked in the inspector — so a single patch can route streams by origin: keyboard into an arp chain, pads into drum lanes, a fader box into CC lanes.

It is a **tap**, not a claim — the **MIDI In** node still receives everything. Needs the input set to **All Inputs** (cable pill). A device that is offline reconnects to its node by name. In a plugin host this node is silent: route the device to the track instead.

#### PARAMETERS

| | |
|---|---|
| `Device` | The hardware input this node taps (inspector dropdown). |

◉ Every parameter accepts amber mod wires (MIDI In signals, LFO, Attractor, Life stats, etc.).



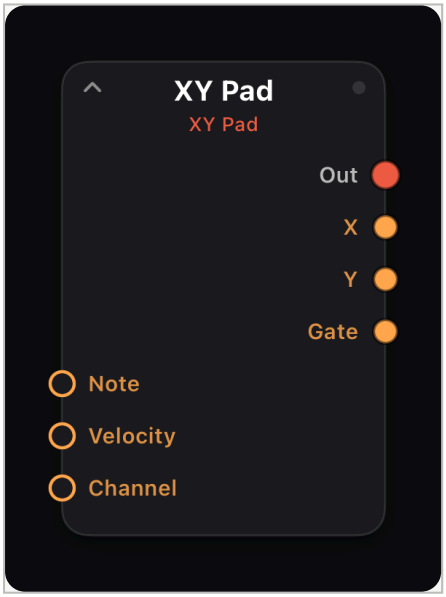

### Keyboard

`In: 1× MIDI (merge) · Out: 1× MIDI`

`Mod out: Velocity · Note · Gate · Mod Wheel · Pitch Bend · Pressure`

The graph node **behind a surface keyboard or pad control** — it emits the notes that control plays, as its **own graph branch**. Every keyboard, pad grid and MPE surface is paired with one, so each can be routed independently: keys into an arp, pads straight to drums.

Adding it from the palette (**Play**) also places a piano keyboard on the surface, pre-wired to **MIDI Out**. Anything wired into **In** merges with the played notes.



### XY Pad

`In: — · Out: 1× MIDI`

`Mod out: X · Y · Gate`

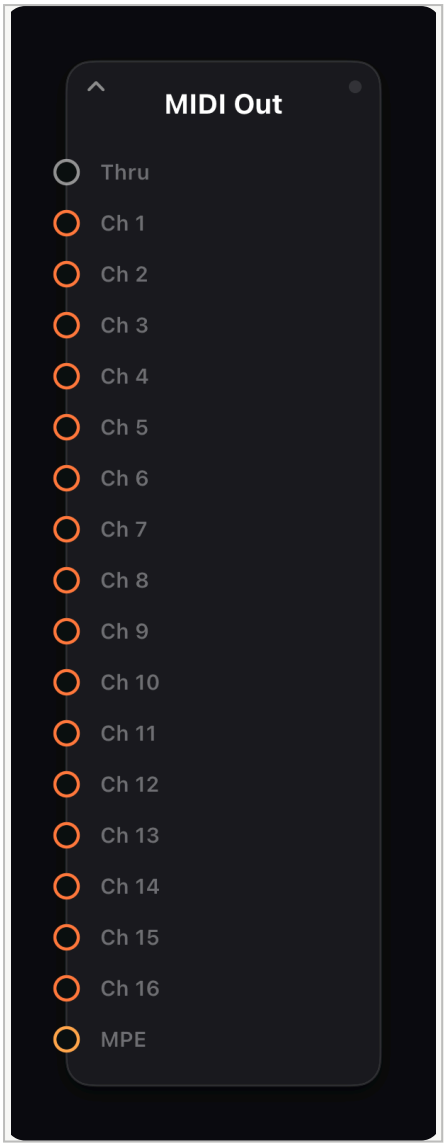
The graph node **behind a surface XY pad**. Wire **Out** into a synth and it plays **Note** for as long as you touch the pad. Wire **X**, **Y** and **Gate** to any parameter. The outputs follow the pad whatever its axes are bound to, and they follow its learned controller and its motion loop, which holds the note and retriggers it at every wrap.

Adding it from the palette (**Play**) also places a pad on the surface, with nothing wired. Deleting the pad deletes the node.

#### PARAMETERS

| | |
|---|---|
| `Note` | MIDI note number 0–127. |
| `Velocity` | Note-on velocity 1–127. |
| `Channel` | MIDI channel 1–16 for the note. |

◉ Every parameter accepts amber mod wires (MIDI In signals, LFO, Attractor, Life stats, etc.).




### MIDI Out

`In: Thru · Ch 1–16 · MPE · Out: — (graph exit)`

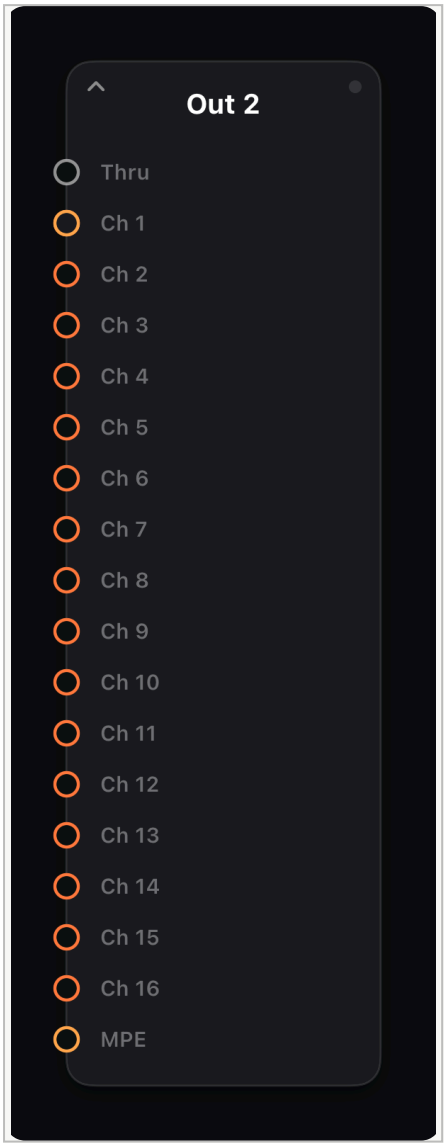
The graph's **exit**. Everything wired here reaches the AUv3 host, the **MIDIRack Out** virtual port, or the built-in synth.

When connecting a wire, choose the input connector that matches your routing goal — see input ports below.

#### INPUT PORTS

| | |
|---|---|
| `Thru` | Merge stream without changing MIDI channel. |
| `Ch 1–16` | Force all output to that channel (multitimbral setups). |

| | |
|---|---|
| MPE | Keep per-note channel assignment for MPE-capable synths. |




**MIDI Out Port**

`In: Thru · Ch 1–16 · MPE · Out: — (graph exit)`

A **second graph exit** on another MIDI output port. Wire a chain into it and it leaves on that port instead of MIDI Out: in a host the plugin shows **MIDIRack Out 2…4** as separate routable outputs, and the standalone publishes matching virtual ports (with their own destination pickers in MIDI Ports).

Fan a patch out to several instruments without channel gymnastics: drums to Out 2, bass to Out 3, each on its own track. One node per port; **MIDI Out** is always port 1. The channel connectors work exactly like MIDI Out's.

#### INPUT PORTS

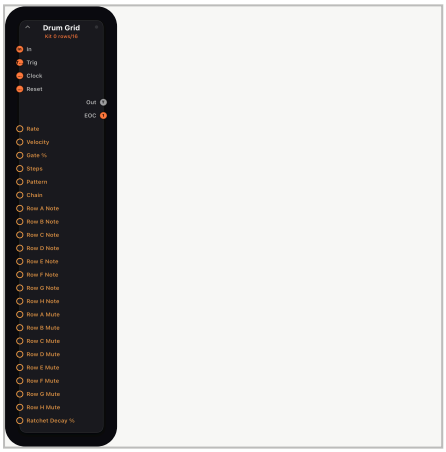
| | |
|---|---|
| `Thru` | Merge stream without changing MIDI channel. |
| `Ch 1–16` | Force all output to that channel (multitimbral setups). |
| `MPE` | Keep per-note channel assignment for MPE-capable synths. |

#### PARAMETERS

| | |
|---|---|
| `Port` | Output port 2–4. |

◉ Every parameter accepts amber mod wires (MIDI In signals, LFO, Attractor, Life stats, etc.).

## Sequence — drum, melody, chord, CC lanes, piano roll, gates, swing




### Drum Grid

`In · Trig · Clock · Reset · Out: 1× MIDI + EOC`

A **whole kit in one sequencer**: eight rows, each with its own **Note** (a GM kit by default — kick, snare, closed and open hat, clap, low tom, crash, ride) and **Mute**, sharing one clock, length and pattern bank. Every row carries the lane layers — per-step velocity, probability, microtiming, ratchets and trigger conditions.

The surface **Drum Grid** control edits all eight rows at once, records pads into their rows, rotates the pattern and switches banks with the rest of the kit. Prefer it over a stack of Drum Seq lanes unless a drum needs its own Chance or length.

### INPUT PORTS

| | |
|---|---|
| `In` | Primary MIDI path — incoming events pass through unchanged to Out. |
| `Trig` | External step clock: each note-on advances the internal pattern one step (overrides **Rate**). |
| `Clock` | Wire from a **Clock** node — one pulse per step (mutually exclusive with **Trig**). |
| `Reset` | A note-on or clock pulse restarts the pattern at step 1 (on either clock mode) — wire a master lane's EOC or a performance pad here to re-align lanes. |

### OUTPUT PORTS

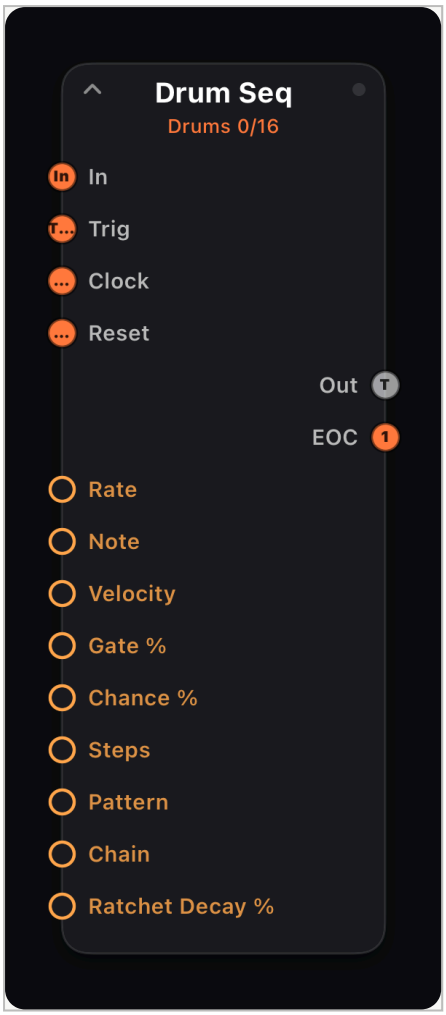
| | |
|---|---|
| `Out` | Every row's drum notes; input passes through alongside. |
| `EOC` | End-of-cycle trigger — one clock pulse each time the pattern wraps to step 1. Wire into another node's **Trig** or **Clock** to chain patterns. |

### PARAMETERS

| | |
|---|---|
| `Rate` | Clock division `4/1` … `1/64`, each with a **dotted** (`1/8D`) and **triplet** (`1/8T`) variant. Follows host transport tempo. |
| `Velocity` | Note-on velocity 1–127. |
| `Gate %` | Note length as percent of one grid step (1–100). |
| `Steps` | Steps per bar, 1–16; a clip runs 1–4 bars of them (its **Bars** menu). |
| `Pattern` | Active pattern bank **A–H** — eight independent clips per lane, each up to four bars long (steps, probability, tie, condition and microtiming all switch together). A switch lands at the lane's launch boundary; pick banks on the Lane+ **A·B·C·D** chips or bind/modulate the parameter. |
| `Chain` | Off, or cycle banks **A·B** / **A·B·C** / **A·B·C·D** automatically — one bank per pattern loop (overrides **Pattern**). |
| `Row A–H Note` | The drum each row plays (GM kit by default). |

| | |
|---|---|
| `Row A–H Mute` | Silences the row without losing its pattern. |
| `Row A–H Step 1–16` | Per-row hits. Drum Grid control only. |
| `Row A–H Vel / Prob / Micro / Ratchet / Cond 1–16` | The lane layers, per row. Drum Grid control only. |
| `Ratchet Decay %` | How far the last retrigger's velocity eases below the first (default 25). |
| `Swing` | **Session** (default) follows the patch's Swing, set on the Rack page's transport row; **Off** plays straight; 1–100 % swings this lane's odd 16ths by its own amount. Adds to Micro; internal clock only. |

◉ Every parameter accepts amber mod wires (MIDI In signals, LFO, Attractor, Life stats, etc.).




### Drum Seq

`In · Trig · Clock · Reset · Out: 1× MIDI + EOC`

A tempo-synced **drum lane** for a single pitch — one node = one drum sound (kick, snare, hat…); a clip is 1–16 steps a bar and up to four bars long. The playhead advances at **Rate** (or from **Trig** / **Clock**); active steps fire note-ons with **Velocity** and **Gate**.

Incoming MIDI on **In** forwards unchanged, so branch several Drum Seq nodes in **parallel** from MIDI In to build a full kit. Bind a surface **Sequencer Lane+** control to edit the pattern along with per-step velocity, probability, tie, and trigger condition — or **record** it: the lane's ● chip takes hits on the lane's **Note** from In (or the patch's MIDI input when In is bare) step by step or live against the running pattern.

#### INPUT PORTS

`In` Primary MIDI path — incoming events pass through unchanged to Out.

| | |
|---|---|
| `Trig` | External step clock: each note-on advances the internal pattern one step (overrides **Rate**). |
| `Clock` | Wire from a **Clock** node — one pulse per step (mutually exclusive with **Trig**). |
| `Reset` | A note-on or clock pulse restarts the pattern at step 1 (on either clock mode) — wire a master lane's EOC or a performance pad here to re-align lanes. |

### OUTPUT PORTS

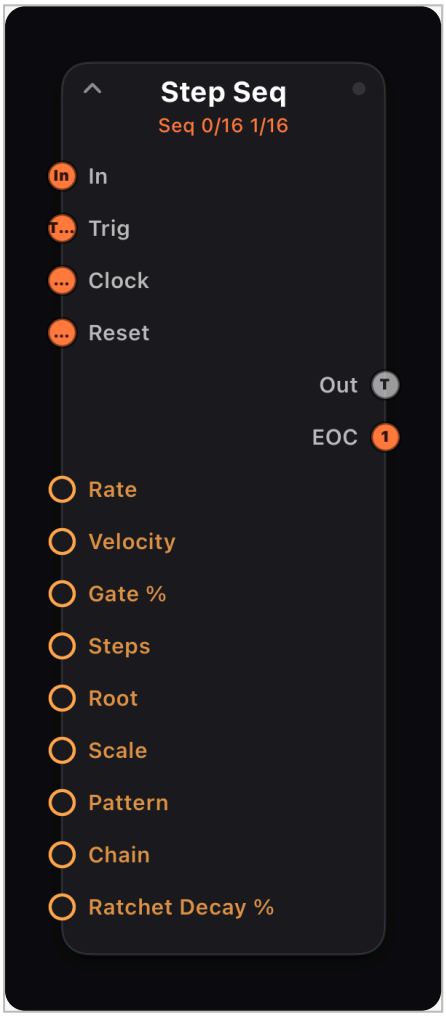
| | |
|---|---|
| `Out` | Drum note-ons; input passes through alongside. |
| `EOC` | End-of-cycle trigger — one clock pulse each time the pattern wraps to step 1. Wire into another node's **Trig** or **Clock** to chain patterns. |

### PARAMETERS

| | |
|---|---|
| `Rate` | Clock division `4/1` … `1/64`, each with a **dotted** (`1/8D`) and **triplet** (`1/8T`) variant. Follows host transport tempo. |
| `Note` | MIDI note number 0–127. |
| `Velocity` | Note-on velocity 1–127. |
| `Gate %` | Note length as percent of one grid step (1–100). |
| `Chance %` | Probability 0–100% that an event fires. |
| `Steps` | Steps per bar, 1–16; a clip runs 1–4 bars of them (its **Bars** menu). |
| `Pattern` | Active pattern bank **A–H** — eight independent clips per lane, each up to four bars long (steps, probability, tie, condition and microtiming all switch together). A switch lands at the lane's launch boundary; pick banks on the Lane+ **A·B·C·D** chips or bind/modulate the parameter. |
| `Chain` | Off, or cycle banks **A·B** / **A·B·C** / **A·B·C·D** automatically — one bank per pattern loop (overrides **Pattern**). |
| `Step 1–16` | Per-step on/off (● = active). Edit on the surface step grid. |
| `Vel 1–16` | Per-step velocity as a percentage of **Velocity** (1–100, default 100) — shape accents and ghost notes. Lane+ control only. |
| `Prob 1–16` | Per-step probability 0–100% (default 100). Lane+ control only. |
| `Tie 1–16` | Per-step tie — suppresses the step's own note and extends the previous note's gate through it instead. Lane+ control only. |

| | |
|---|---|
| `Cond 1–16` | Per-step trigger condition — fires only on that one pass out of every *M* loops (Always, `1:2`…`4:4`). Lane+ control only. |
| `Micro 1–16` | Per-step timing nudge –50% … +50% of one grid step. Surface grid only. |
| `Ratchet 1–16` | Per-step ratchet ×1 (default) … ×4: the step's note retriggers that many times, evenly inside the step, each note-off landing before the next hit. Lane+ control only. |
| `Ratchet Decay %` | How far the last retrigger's velocity eases below the first (default 25). |
| `Swing` | **Session** (default) follows the patch's Swing, set on the Rack page's transport row; **Off** plays straight; 1–100 % swings this lane's odd 16ths by its own amount. Adds to Micro; internal clock only. |

◉ Every parameter accepts amber mod wires (MIDI In signals, LFO, Attractor, Life stats, etc.).



### Step Seq

`In · Trig · Clock · Reset · Out: 1× MIDI + EOC`

A **monophonic melody sequencer**: one pitch per step (0 = rest). The playhead steps at **Rate** or external clock with configurable **Gate** and **Velocity**.

Built-in **Root** + **Scale** snap each step's output in-key (**Chromatic** = off). Draw pitches on the surface **Sequencer Lane+** control — the ladder shows the sounding note after quantization, and a layer picker switches to per-step velocity, probability, tie, and trigger condition. Or **record** the line: the lane's ● chip takes notes from In (or the patch's MIDI input when In is bare) step by step or live, held notes tying the steps they span.

#### INPUT PORTS

| | |
|---|---|
| `In` | Primary MIDI path — incoming events pass through unchanged to Out. |
| `Trig` | External step clock: each note-on advances the internal pattern one step (overrides **Rate**). |
| `Clock` | Wire from a **Clock** node — one pulse per step (mutually exclusive with **Trig**). |
| `Reset` | A note-on or clock pulse restarts the pattern at step 1 (on either clock mode) — wire a master lane's EOC or a performance pad here to re-align lanes. |

#### OUTPUT PORTS

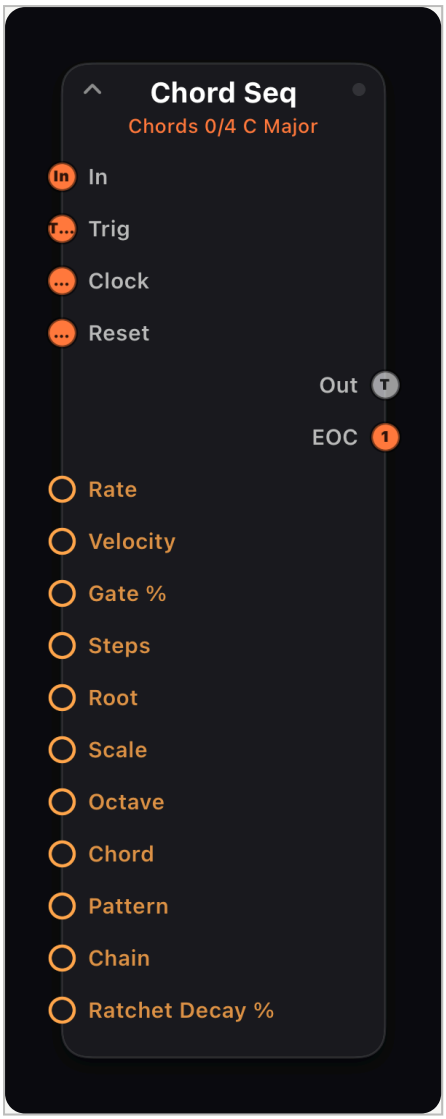
| | |
|---|---|
| `Out` | Melody note per step; input passes through alongside. |
| `EOC` | End-of-cycle trigger — one clock pulse each time the pattern wraps to step 1. Wire into another node's **Trig** or **Clock** to chain patterns. |

#### PARAMETERS

| | |
|---|---|
| `Rate` | Clock division `4/1` … `1/64`, each with a **dotted** (`1/8D`) and **triplet** (`1/8T`) variant. Follows host transport tempo. |
| `Velocity` | Note-on velocity 1–127. |
| `Gate %` | Note length as percent of one grid step (1–100). |
| `Steps` | Steps per bar, 1–16; a clip runs 1–4 bars of them (its **Bars** menu). |
| `Root` | Scale root (C, C#, D…). Used with **Scale** to snap output notes in-key. |
| `Scale` | Scale name (Major, Minor, Pentatonic…, or one of the patch's **custom scales**). **Chromatic** (default) passes every note unchanged — effectively off. |

| | |
|---|---|
| `Pattern` | Active pattern bank **A–H** — eight independent clips per lane, each up to four bars long (steps, probability, tie, condition and microtiming all switch together). A switch lands at the lane's launch boundary; pick banks on the Lane+ **A·B·C·D** chips or bind/modulate the parameter. |
| `Chain` | Off, or cycle banks **A·B** / **A·B·C** / **A·B·C·D** automatically — one bank per pattern loop (overrides **Pattern**). |
| `Step 1–16` | MIDI note per step (0 = rest). Edit on the surface step grid. |
| `Vel 1–16` | Per-step velocity as a percentage of **Velocity** (1–100, default 100) — shape accents and ghost notes. Lane+ control only. |
| `Prob 1–16` | Per-step probability 0–100% (default 100). Lane+ control only. |
| `Tie 1–16` | Per-step tie — suppresses the step's own note and extends the previous note's gate through it instead. Lane+ control only. |
| `Cond 1–16` | Per-step trigger condition — fires only on that one pass out of every *M* loops (Always, `1:2`…`4:4`). Lane+ control only. |
| `Micro 1–16` | Per-step timing nudge –50% … +50% of one grid step. Surface grid only. |
| `Ratchet 1–16` | Per-step ratchet ×1 (default) … ×4: the step's note retriggers that many times, evenly inside the step, each note-off landing before the next hit. Lane+ control only. |
| `Ratchet Decay %` | How far the last retrigger's velocity eases below the first (default 25). |
| `Swing` | **Session** (default) follows the patch's Swing, set on the Rack page's transport row; **Off** plays straight; 1–100 % swings this lane's odd 16ths by its own amount. Adds to Micro; internal clock only. |

◉ Every parameter accepts amber mod wires (MIDI In signals, LFO, Attractor, Life stats, etc.).




### Chord Seq

`In · Trig · Clock · Reset · Out: 1× MIDI + EOC`

A **chord progression sequencer**: each step plays a diatonic chord on a scale degree (0 = rest, 1–7 = I…VII), voiced by stacking thirds in **Root** + **Scale** at **Octave** — as triads or, with **Chord** set to 7th, four-voice sevenths. Degrees 1·6·4·5 in C Major play the classic C–Am–F–G.

Same step machinery as the other lanes: the surface **Sequencer Lane+** control shows each step as a **named chord pad** ("C", "Am", "G7" with its roman numeral) that follows the Root / Scale / Chord knobs live, with per-step velocity and probability layers. Under each pad a **Tie** button holds the previous chord through the step.

The lane's **Rhythm** layer is a second grid with its own **Rhythm Rate** and **Rhythm Steps**: while any of its hits is set, the chord steps only *choose* the chord and the rhythm grid plays it — one chord per bar on the progression, stabs on 2 and 4 (or every offbeat eighth) on the rhythm.

Each hit's height is its accent; Gate % applies to the rhythm step. An empty rhythm leaves every chord step sounding on its own.

The voicing button under each pad sets the step's inversion — Root, 1st, 2nd or 3rd, each lifting the lowest note an octave (a layer of its own on a lane too narrow for the buttons) — so a progression can be voice-led instead of jumping between root positions; the Chords player writes it for you.

**Strum** rolls a step's chord instead of landing it as a block: the voices start one after another over the step's Strum, a share of the step (so the roll scales with Rate), **Up** from the lowest sounding note, **Down** from the highest, **Alternate** per strum or **Random**, spaced **Even**, **Tight** (a flick) or **Loose**; the notes lift together right after the roll, so the instrument's release rings them out (a tie holds a strum on). Rhythm hits roll over their own step and every ratchet rolls again. The lane's **Strum** layer sets every step's roll, with the direction and curve on chips beside it, so one hard strum on the one can sit among tight pads. The Chords player strums every chord to suit its feel.

### INPUT PORTS

| | |
|---|---|
| `In` | Primary MIDI path — incoming events pass through unchanged to Out. |
| `Trig` | External step clock: each note-on advances the internal pattern one step (overrides **Rate**). |
| `Clock` | Wire from a **Clock** node — one pulse per step (mutually exclusive with **Trig**). |
| `Reset` | A note-on or clock pulse restarts the pattern at step 1 (on either clock mode) — wire a master lane's EOC or a performance pad here to re-align lanes. |

### OUTPUT PORTS

| | |
|---|---|
| `Out` | Chord tones per step; input passes through alongside. |
| `EOC` | End-of-cycle trigger — one clock pulse each time the pattern wraps to step 1. Wire into another node's **Trig** or **Clock** to chain patterns. |

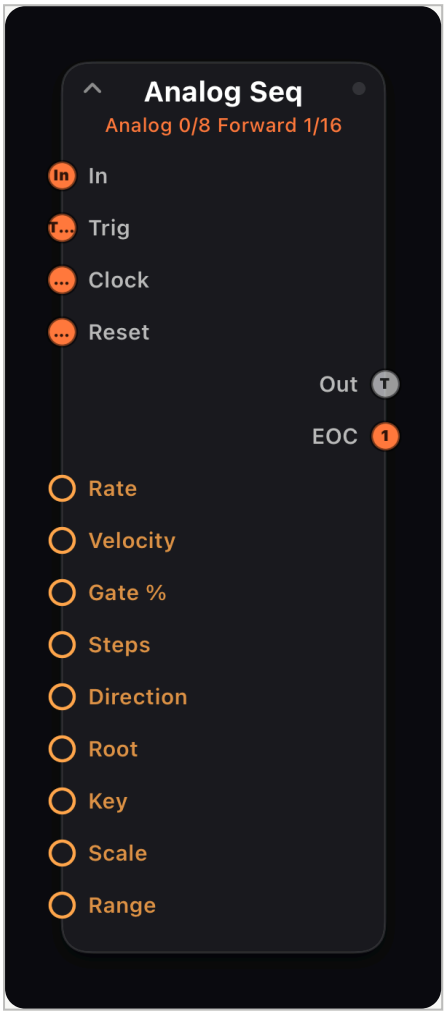
### PARAMETERS

| | |
|---|---|
| `Rate` | Clock division `4/1` … `1/64`, each with a **dotted** (`1/8D`) and **triplet** (`1/8T`) variant. Follows host transport tempo. |
| `Velocity` | Note-on velocity 1–127. |
| `Gate %` | Note length as percent of one grid step (1–100). |
| `Steps` | Steps per bar, 1–16; a clip runs 1–4 bars of them (its **Bars** menu). |
| `Root` | Key root (C, C#, D…) the degrees are built in. |

| | |
|---|---|
| `Scale` | Scale the chords are stacked from (Major, Minor, modes…, or a custom scale). |
| `Octave` | Register of the degree-1 chord root (1–7). |
| `Chord` | Triad or 7th — three or four stacked scale thirds. |
| `Pattern` | Active pattern bank **A–H** — eight independent clips per lane, each up to four bars long (steps, probability, tie, condition and microtiming all switch together). A switch lands at the lane's launch boundary; pick banks on the Lane+ **A·B·C·D** chips or bind/modulate the parameter. |
| `Chain` | Off, or cycle banks **A·B** / **A·B·C** / **A·B·C·D** automatically — one bank per pattern loop (overrides **Pattern**). |
| `Step 1–16` | Scale degree per step (0 = rest, 1–7 = I…VII). Edit on the surface step grid. |
| `Vel 1–16` | Per-step velocity as a percentage of **Velocity** (1–100, default 100) — shape accents and ghost notes. Lane+ control only. |
| `Prob 1–16` | Per-step probability 0–100% (default 100). Lane+ control only. |
| `Tie 1–16` | Per-step tie — suppresses the step's own note and extends the previous note's gate through it instead. Lane+ control only. |
| `Cond 1–16` | Per-step trigger condition — fires only on that one pass out of every *M* loops (Always, `1:2`…`4:4`). Lane+ control only. |
| `Micro 1–16` | Per-step timing nudge –50% … +50% of one grid step. Surface grid only. |
| `Ratchet 1–16` | Per-step ratchet ×1 (default) … ×4: the step's note retriggers that many times, evenly inside the step, each note-off landing before the next hit. Lane+ control only. |
| `Ratchet Decay %` | How far the last retrigger's velocity eases below the first (default 25). |
| `Rhythm Rate` | Grid of the rhythm layer (default 1/16). |
| `Rhythm Steps` | Length of the rhythm grid, 1–16 (default 16). |
| `Rhythm 1–16` | Rhythm hits, accent 0–100 % (0 = no hit). Any hit set hands the triggering to this grid. Banked with the chord steps, so each clip has its own rhythm. Lane+ control only. |
| `Strum Direction` | Up (low to high), Down, Alternate per strum, or Random. |
| `Strum Curve` | Even gaps, Tight (closing toward the top, a flick) or Loose (opening). |

| | |
|---|---|
| `Strum 1–16` | The step's roll, 0–100 % of the step (0 = the block chord). Banked with the chord steps. Lane+ control only. |
| `Swing` | **Session** (default) follows the patch's Swing, set on the Rack page's transport row; **Off** plays straight; 1–100 % swings this lane's odd 16ths by its own amount. Adds to Micro; internal clock only. |

◉ Every parameter accepts amber mod wires (MIDI In signals, LFO, Attractor, Life stats, etc.).




### Analog Seq

`In · Trig · Clock · Reset · Out: 1× MIDI + EOC`

The **knob-per-step analog machine**: eight pitches set as semitone offsets above **Root** (0 = off, +0 up to the row's **Range** of 1, 2 or 3 octaves), walked **Forward, Backward, Pendulum or Random**. No banks, no piano roll — one row you dial in and perform.

**Glide** downstream and slides become true 303-style portamento. The velocity layer is the **accent row**: drag steps up to punch them.

### INPUT PORTS

| | |
|---|---|
| `In` | Primary MIDI path — incoming events pass through unchanged to Out. |
| `Trig` | External step clock: each note-on advances the internal pattern one step (overrides **Rate**). |
| `Clock` | Wire from a **Clock** node — one pulse per step (mutually exclusive with **Trig**). |
| `Reset` | A note-on or clock pulse restarts the pattern at step 1 (on either clock mode) — wire a master lane's EOC or a performance pad here to re-align lanes. |

### OUTPUT PORTS

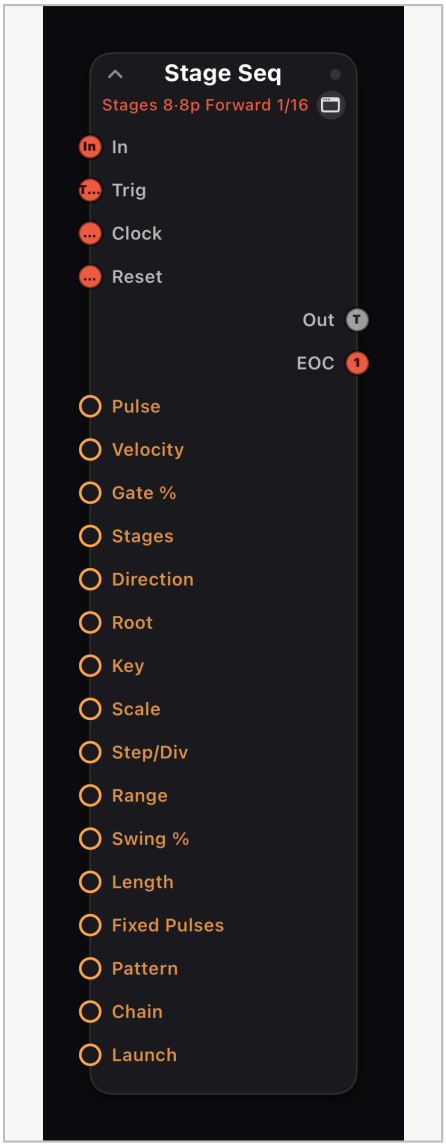
| | |
|---|---|
| `Out` | The stepped row; non-note input passes through alongside. |
| `EOC` | End-of-cycle trigger — one clock pulse each time the pattern wraps to step 1. Wire into another node's **Trig** or **Clock** to chain patterns. |

### PARAMETERS

| | |
|---|---|
| `Rate` | Clock division `4/1` … `1/64`, each with a **dotted** (`1/8D`) and **triplet** (`1/8T`) variant. Follows host transport tempo. |
| `Velocity` | Note-on velocity 1–127. |
| `Gate %` | Note length as percent of one grid step (1–100). |
| `Steps` | Row length, 1–8. |
| `Direction` | Forward · Backward · Pendulum · Random walk of the row. |
| `Root` | The row's octave when no key is held — the root itself is the Key's tonic in that octave, so a row in A starts on an A. Playing a note re-roots the row. |
| `Key` | The row's tonic and quantizer root; sounding pitches snap to Key + Scale. |
| `Scale` | Quantizer scale, built-in or custom (Chromatic = off — offsets pass unchanged). |
| `Step 1–8` | Semitone offset per step (0 = off, 1–37 = +0…+36 st, capped by **Range**). Edit on the surface step grid. |
| `Vel 1–8` | Accent row — per-step % of Velocity. |
| `Slide 1–8` | Hold this step through the next onset (legato tie). |

| | |
|---|---|
| `Cond 1–8` | Per-step trigger condition — fires only on that one pass out of every *M* walk cycles (Always, `1:2`…`4:4`). Lane+ control only. |
| `Ratchet 1–8` | Per-step ratchet ×1 (default) … ×4: the knob's note retriggers that many times inside the step; on a slide step the last retrigger holds through. Lane+ control only. |
| `Range` | Octave span of the pitch knobs — 1, 2 (default) or 3 octaves; offsets above it are capped. |
| `Ratchet Decay %` | How far the last retrigger's velocity eases below the first (default 25). |
| `Swing` | **Session** (default) follows the patch's Swing, set on the Rack page's transport row; **Off** plays straight; 1–100 % swings this lane's odd 16ths by its own amount. Adds to Micro; internal clock only. |

◉ Every parameter accepts amber mod wires (MIDI In signals, LFO, Attractor, Life stats, etc.).




### Stage Seq

`In · Trig · Clock · Reset · Out: 1× MIDI + EOC`

A **stage-and-pulse sequencer** in the Intellijel Metropolis mould. Eight **stages**, each a pitch that lasts **1–8 clock pulses** — the rhythm comes from the pulse counts, and the loop is their sum (3·1·2·2 is an eight-pulse riff with that feel). Each stage has a **gate mode** for its pulses: **Rest** (silent), **Single** (one gate on the first pulse), **Repeat** (a gate every **Step/Div** pulses) or **Hold** (one note for the whole stage).

**Slide** makes a stage arrive legato — the note before holds through its onset; put **Mono + Glide** downstream for real portamento. **Skip** removes a stage without touching its settings; a **Ratchet** splits each of its gates into 2–4. Walk **Forward, Reverse, Ping-Pong, Brownian** (a

roots to it live, like the Analog Seq. Eight pattern banks, launched from Scene Pads like every lane.

### INPUT PORTS

| | |
|---|---|
| `In` | Primary MIDI path — incoming events pass through unchanged to Out. |
| `Trig` | External step clock: each note-on advances the internal pattern one step (overrides **Rate**). |
| `Clock` | Wire from a **Clock** node — one pulse per step (mutually exclusive with **Trig**). |
| `Reset` | A note-on or clock pulse restarts the pattern at step 1 (on either clock mode) — wire a master lane's EOC or a performance pad here to re-align lanes. |

### OUTPUT PORTS

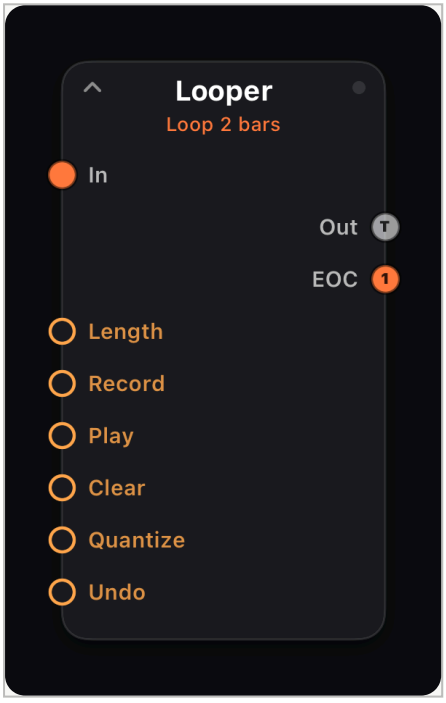
| | |
|---|---|
| `Out` | The stage row; non-note input passes through alongside. |
| `EOC` | End-of-cycle trigger — one clock pulse each time the pattern wraps to step 1. Wire into another node's **Trig** or **Clock** to chain patterns. |

### PARAMETERS

| | |
|---|---|
| `Pulse` | The clock pulse — a stage lasts 1–8 of them. |
| `Velocity` | Note-on velocity 1–127. |
| `Gate %` | Gate length as a % of a pulse (Single, Repeat). |
| `Stages` | How many stages are in play, 1–8 (tap a stage number on the panel). |
| `Direction` | Forward · Reverse · Ping-Pong · Brownian · Random. |
| `Root` | The row's octave when no key is held (the Key's tonic in that octave). Playing a note re-roots the row. |
| `Key` | The row's tonic and quantizer root. |
| `Scale` | Quantizer scale (Chromatic = off). |
| `Step/Div` | Repeat stages fire a gate every 1–4 pulses. |
| `Range` | Span of the pitch sliders — 1, 2 (default) or 3 octaves. |
| `Swing` | **Off** (50, straight) … 75 %: every second pulse lands later. **Session**, below Off, follows the patch's Swing on the odd 16ths instead — where a new lane starts. |
| `Length` | Stages (the cycle is the stages' pulses) or Fixed (it restarts every Fixed Pulses). |

| | |
|---|---|
| `Fixed Pulses` | Cycle length in Fixed mode, 1–64 pulses. |
| `Pitch 1–8` | Semitones above the root, 0–36 (capped by Range). |
| `Pulses 1–8` | How many pulses the stage lasts, 1–8. |
| `Gate 1–8` | Rest (default) · Single · Repeat · Hold. |
| `Slide 1–8` | Arrive at this stage legato. |
| `Skip 1–8` | Leave the stage out — it takes no time. |
| `Ratchet 1–8` | Split each gate into 1–4 retriggers. |

◉ Every parameter accepts amber mod wires (MIDI In signals, LFO, Attractor, Life stats, etc.).




### Piano Roll

`In: 1× MIDI · Out: 1× MIDI + EOC`

Tempo-synced **MIDI phrase recorder** whose recording is saved with the patch: **Record** overdubs live notes *and* expression (CC, bend, pressure) as a new take, **Play** loops what is captured, **Undo** drops the newest take, **Clear** wipes the buffer. **Length** sets the loop size in bars.

**Quantize** snaps recorded note-ons to a grid (Off keeps your timing); note-offs and expression move with their note, so durations and MPE gestures keep their shape. Drive from a surface Piano Roll card for hands-free capture.

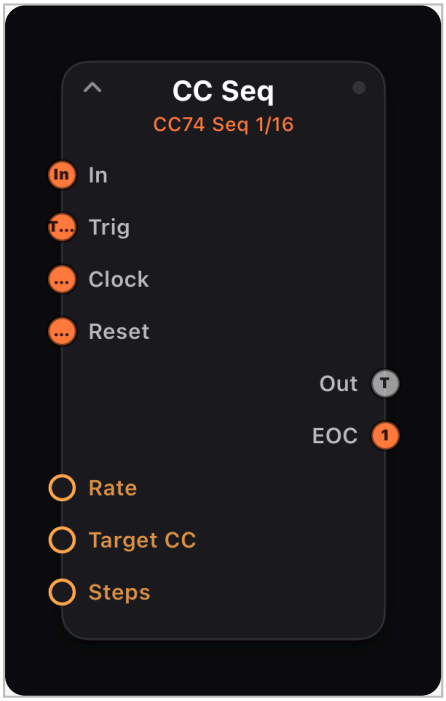
### OUTPUT PORTS

| | |
|---|---|
| `Out` | Live input plus looped playback. |
| `EOC` | End-of-cycle trigger — one clock pulse each time the pattern wraps to step 1. Wire into another node's **Trig** or **Clock** to chain patterns. |

### PARAMETERS

| | |
|---|---|
| `Length` | Loop length in bars 1–8. |
| `Record` | Off / On — overdub live input as a new take. |
| `Play` | Off / On — playback loop. |
| `Clear` | Momentary — wipe loop buffer. |
| `Quantize` | Off / 1/4 ... 1/32 — snap recorded note-ons to the grid. |
| `Undo` | Momentary — drop the newest take. |

◉ Every parameter accepts amber mod wires (MIDI In signals, LFO, Attractor, Life stats, etc.).



### CC Seq

`In · Trig · Clock · Reset · Out: 1× MIDI + EOC`

A **16-step automation lane** for one CC number — stepped filter sweeps, acid resonance, rhythmic tremolo. Each clock step writes the matching step value on **Target CC** while note data passes through.

Layer several CC Seq nodes on parallel branches to automate multiple parameters in sync.

#### INPUT PORTS

| | |
|---|---|
| `In` | Primary MIDI path — incoming events pass through unchanged to Out. |
| `Trig` | External step clock: each note-on advances the internal pattern one step (overrides **Rate**). |
| `Clock` | Wire from a **Clock** node — one pulse per step (mutually exclusive with **Trig**). |
| `Reset` | A note-on or clock pulse restarts the pattern at step 1 (on either clock mode) — wire a master lane's EOC or a performance pad here to re-align lanes. |

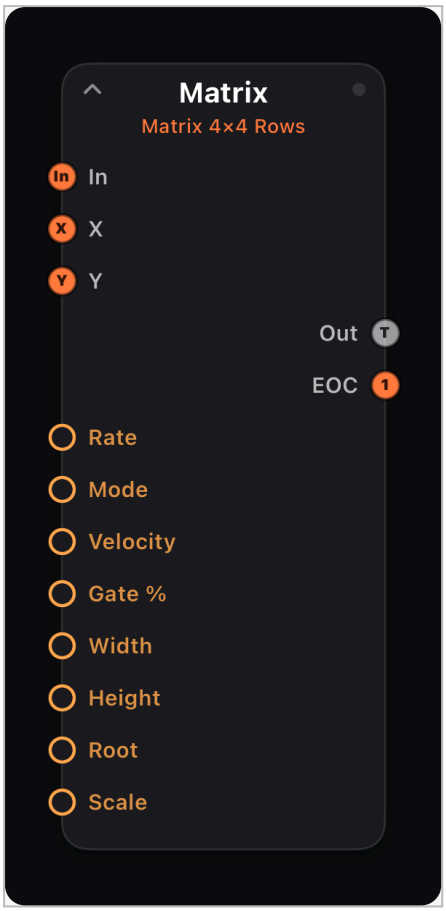
#### OUTPUT PORTS

| | |
|---|---|
| `Out` | MIDI passthrough with the stepped value written on **Target CC**. |
| `EOC` | End-of-cycle trigger — one clock pulse each time the pattern wraps to step 1. Wire into another node's **Trig** or **Clock** to chain patterns. |

#### PARAMETERS

| | |
|---|---|
| `Rate` | Clock division `4/1` … `1/64`, each with a **dotted** (`1/8D`) and **triplet** (`1/8T`) variant. Follows host transport tempo. |
| `Target CC` | MIDI CC number to write (0–127). |
| `Steps` | Pattern length 1–16. |
| `Step 1–16` | CC value 0–127 per step. Edit on the surface step grid. |
| `Prob 1–16` | Per-step probability 0–100% (default 100). Lane+ control only. |
| `Cond 1–16` | Per-step trigger condition — fires only on that one pass out of every *M* loops (Always, `1:2`…`4:4`). Lane+ control only. |

◉ Every parameter accepts amber mod wires (MIDI In signals, LFO, Attractor, Life stats, etc.).




### Matrix

`In · X Trig · Y Trig · Out: 1× MIDI + EOC`

A **grid sequencer**: a grid of note values (0 = rest, default 4×4, up to 8×8). The internal clock walks the grid at **Rate** in a **Mode** — Rows, Columns, Snake, Diagonal, Spiral or Random — one cell per step, playing the current cell's note.

Or scan it as an **XY sequencer**: wire clocks into **X Trig** and **Y Trig** to advance column and row independently. Built-in **Root** + **Scale** snap each cell's output in-key (**Chromatic** = off). Like the other sequencers it pulses **EOC** when the cursor returns to the origin. Bind a surface **Matrix** control to edit the grid with a live playhead.

#### INPUT PORTS

| | |
|---|---|
| `In` | Primary MIDI path — passes through unchanged to Out. |
| `X Trig` | Note-ons advance the **column** one step (overrides the internal clock). |
| `Y Trig` | Note-ons advance the **row** one step — wire two clocks for XY scanning. |

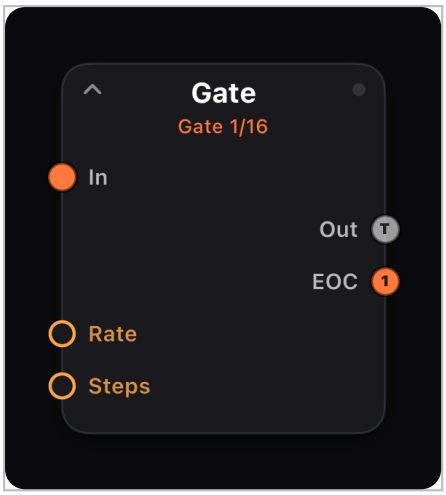
#### OUTPUT PORTS

| | |
|---|---|
| `Out` | Note of the current cell (0 = rest); input passes through. |
| `EOC` | End-of-cycle trigger — one clock pulse each time the pattern wraps to step 1. Wire into another node's **Trig** or **Clock** to chain patterns. |

### PARAMETERS

| | |
|---|---|
| `Rate` | Clock division `4/1` … `1/64`, each with a **dotted** (`1/8D`) and **triplet** (`1/8T`) variant. Follows host transport tempo. |
| `Mode` | Traversal: Rows, Columns, Snake, Diagonal, Spiral, Random. |
| `Velocity` | Note-on velocity 1–127. |
| `Gate %` | Note length as percent of one grid step (1–100). |
| `Width` | Active grid columns 2–8. |
| `Height` | Active grid rows 2–8. |
| `Root` | Scale root (C, C#, D…). Used with **Scale** to snap output notes in-key. |
| `Scale` | Scale name (Major, Minor, Pentatonic…, or one of the patch's **custom scales**). **Chromatic** (default) passes every note unchanged — effectively off. |
| `Cell 1–64` | MIDI note per cell (0 = rest); the active Width × Height subset plays. Edit on the surface Matrix grid. |
| `Prob 1–64` | Per-cell probability 0–100% (default 100). Engine-only — no dedicated surface control yet, but wireable as a mod destination. |
| `Tie 1–64` | Per-cell tie — suppresses the cell's own note and extends the previously fired note's gate through it instead. Engine-only. |
| `Cond 1–64` | Per-cell trigger condition — fires only on that one pass out of every *M* loops around the origin (Always, `1:2`…`4:4`). Engine-only. |

◉ Every parameter accepts amber mod wires (MIDI In signals, LFO, Attractor, Life stats, etc.).




### Gate

`In: 1× MIDI · Out: 1× MIDI + EOC`

Also called **Trance Gate**. **Holds** incoming notes and **chops** them with a 16-step on/off pattern — sustained chords become a rhythmic pulse without retriggering the source.

Unlike a sequencer, it reacts to whatever you play in real time. Draw the gate pattern on the step row for trance, house and EDM stutters.

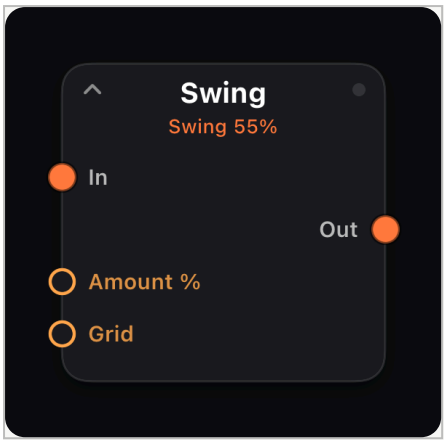
#### OUTPUT PORTS

| | |
|---|---|
| `Out` | Gated note stream. |
| `EOC` | End-of-cycle trigger — one clock pulse each time the pattern wraps to step 1. Wire into another node's **Trig** or **Clock** to chain patterns. |

#### PARAMETERS

| | |
|---|---|
| `Rate` | Clock division `4/1` … `1/64`, each with a **dotted** (`1/8D`) and **triplet** (`1/8T`) variant. Follows host transport tempo. |
| `Steps` | Pattern length 1–16. |
| `Step 1–16` | Gate open (1) or closed (0) per grid step. Draw on the step row. |

◉ Every parameter accepts amber mod wires (MIDI In signals, LFO, Attractor, Life stats, etc.).




### Swing

`In: 1× MIDI · Out: 1× MIDI`

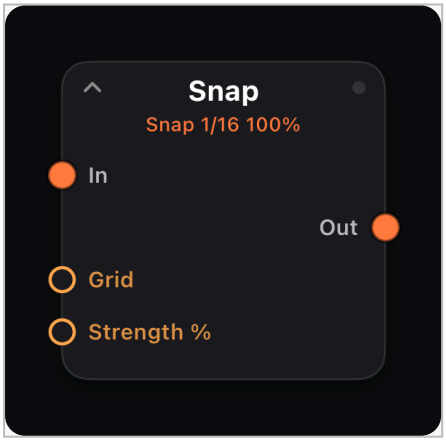
Adds **shuffle** by delaying off-beat note-ons. **Amount** sets how late (percent of half a grid step); **Grid** is the timing reference.

Place after sequencers, arps or generators to loosen rigid timing without changing the underlying pattern.

#### PARAMETERS

| | |
|---|---|
| `Amount %` | Shuffle depth 0–100%. Higher = more laid-back off-beats. |
| `Grid` | Reference clock division for quantization or swing (`4/1` ... `1/64`, dotted and triplet variants included). |

◉ Every parameter accepts amber mod wires (MIDI In signals, LFO, Attractor, Life stats, etc.).



### Snap

`In: 1× MIDI · Out: 1× MIDI`

**Quantizes note-on timing** to the clock grid while preserving note lengths. **Strength** blends dry input (0%) with fully locked timing (100%).

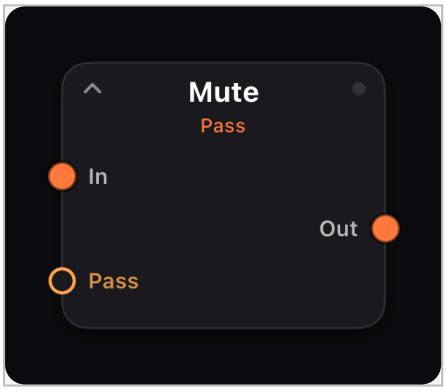
Use after live keyboard input or **Humanize** to tighten a part without reprogramming it.

PARAMETERS

| | |
|---|---|
| `Grid` | Reference clock division for quantization or swing (`4/1` … `1/64`, dotted and triplet variants included). |
| `Strength %` | Blend 0% = passthrough timing, 100% = fully snapped. |

◉ Every parameter accepts amber mod wires (MIDI In signals, LFO, Attractor, Life stats, etc.).

## Control — mute, A/B switch, seq router, triggers




### Mute

`In: 1× MIDI · Out: 1× MIDI`

Branch **mute / bypass**. When **Pass** is Off the stream is silenced and held notes are flushed; On lets events through normally.

Bind a toggle button to **Pass** to mute a part during performance.

PARAMETERS

| | |
|---|---|
| `Pass` | Off = muted (notes flushed), On = live. |

◉ Every parameter accepts amber mod wires (MIDI In signals, LFO, Attractor, Life stats, etc.).




### A/B Switch

`In: A · B · Trig · Out: 1× MIDI`

Routes one of two musical inputs to **Out**. **Active** picks A or B when no trigger is active. Wire a sequencer or keyboard to **Trig** for performance switching.

In **Gate** mode (default), B is selected while trigger notes are held; **Toggle** flips on each trigger note-on. **Off** ignores Trig.

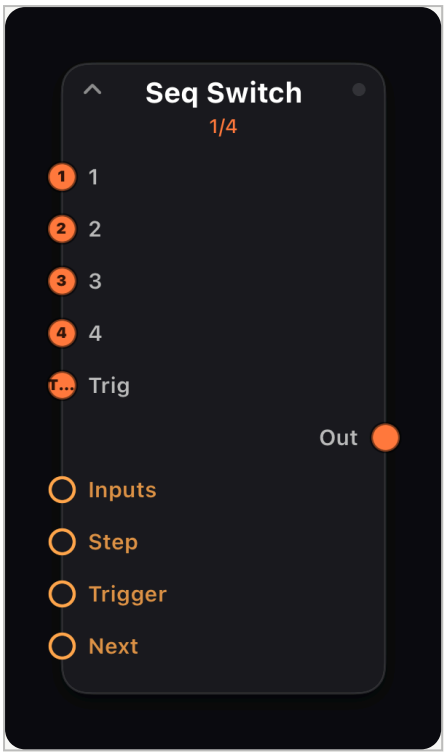
#### INPUT PORTS

| | |
|---|---|
| `A` | First musical input — heard when **Active** is A. |
| `B` | Second musical input — heard when **Active** is B. |
| `Trig` | Control input — note-ons here drive **Gate** (hold B while trigger notes play) or **Toggle** mode. |

#### PARAMETERS

| | |
|---|---|
| `Active` | A or B when trigger input is idle. |
| `Trigger` | Off / Gate / Toggle — how **Trig** port behaves. |

◉ Every parameter accepts amber mod wires (MIDI In signals, LFO, Attractor, Life stats, etc.).




### Seq Switch

`In: 1…N + Trig · Out: 1× MIDI`

**Sequential router**: wire different musical sources to inputs **1…N** and a clock or trigger source to **Trig**. Each trigger note-on advances the active step and passes only that input through.

**Step** selects manually when **Trigger** is Off. Held notes are flushed on step change to avoid stuck notes.

#### INPUT PORTS

| | |
|---|---|
| `1…N` | Musical inputs (count = **Inputs**). Only the active step's stream reaches Out. |
| `Trig` | Advance clock — each note-on moves to the next input when **Trigger** is On. |

#### PARAMETERS

| | |
|---|---|
| `Inputs` | Number of musical input ports (2–8). |
| `Step` | Active input index (0-based). Manual when Trigger is Off. |
| `Trigger` | Off = manual **Step**; On = advance on each Trig note-on. |
| `Next` | Momentary — advance one step (same as a trigger pulse). |

◉ Every parameter accepts amber mod wires (MIDI In signals, LFO, Attractor, Life stats, etc.).




### Pulse

`In: 1× MIDI · Out: 1× MIDI`

**Momentary trigger** node. A rising edge on **Fire** emits a short note burst or a CC value depending on **Mode**; in **Gate** mode the note holds until **Fire** goes low. Incoming MIDI passes through unchanged.

Bind a momentary pad or button to **Fire** for one-shot hits and automation spikes, or for a note that holds while pressed in **Gate** mode.

#### PARAMETERS

| | |
|---|---|
| `Fire` | Momentary — rising edge fires the pulse. |
| `Mode` | Note = MIDI note burst; CC = single CC value; Gate = note held while Fire is on. |
| `Note` | MIDI note number 0–127. |
| `Velocity` | Note-on velocity 1–127. |
| `Channel` | MIDI channel 1–16 for Note and Gate modes. |
| `CC` | CC number for CC mode. |
| `Value` | CC value 0–127 for CC mode. |

◉ Every parameter accepts amber mod wires (MIDI In signals, LFO, Attractor, Life stats, etc.).

# Filter & Route — channel, range, split, chance




### Channel

`In: 1× MIDI · Out: 1× MIDI`

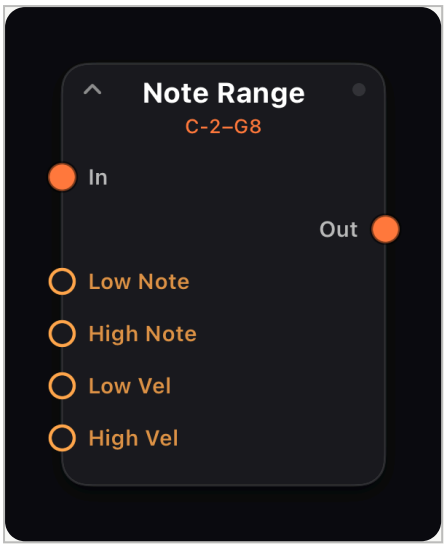
Filters MIDI by channel and optionally remaps it. Passes only events on **Input Ch** (0 = any channel) and can rewrite them to **Output Ch** (0 = leave channel unchanged).

Route one branch to channel 10 for drums and another to channel 1 for bass in a multitimbral rig.

PARAMETERS

| | |
|---|---|
| `Input Ch` | 0 = accept any channel; 1–16 = filter to that channel. |
| `Output Ch` | 0 = keep original channel; 1–16 = force output channel. |

◉ Every parameter accepts amber mod wires (MIDI In signals, LFO, Attractor, Life stats, etc.).




### Note Range

`In: 1× MIDI · Out: 1× MIDI`

Passes only notes inside pitch and velocity windows. Events outside **Low/High Note** or **Low/High Vel** are dropped; CC and other message types pass through.

Split a keyboard by pitch, isolate hard hits, or block low-end rumble before a bass synth.

#### PARAMETERS

| | |
|---|---|
| `Low Note` | Lowest pitch in range (0–127). |
| `High Note` | Highest pitch in range (0–127). |
| `Low Vel` | Minimum velocity 1–127. |
| `High Vel` | Maximum velocity 1–127. |

◉ Every parameter accepts amber mod wires (MIDI In signals, LFO, Attractor, Life stats, etc.).




### Msg Filter

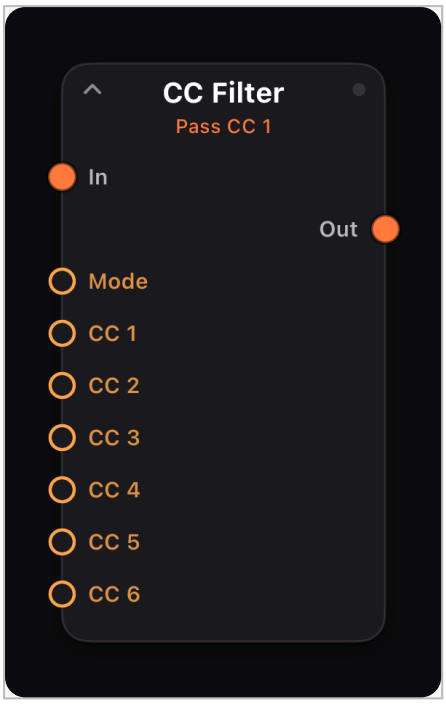
`In: 1× MIDI · Out: 1× MIDI`

Per-type **Pass/Block** switches for Notes, CC, Pitch Bend, Aftertouch and Program Change. Strip CC clutter or pass only notes to a monophonic line.

#### PARAMETERS

| | |
|---|---|
| `Notes` | Pass or Block note on/off. |
| `CC` | Pass or Block control change. |
| `Pitch Bend` | Pass or Block pitch wheel. |
| `Aftertouch` | Pass or Block channel pressure. |
| `Prog Change` | Pass or Block program change. |

◉ Every parameter accepts amber mod wires (MIDI In signals, LFO, Attractor, Life stats, etc.).




### CC Filter

`In: 1× MIDI · Out: 1× MIDI`

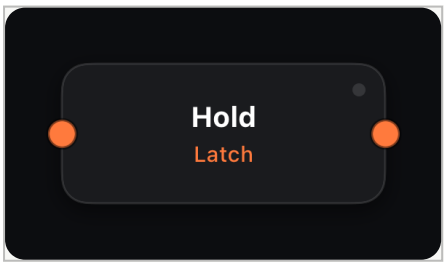
Per-**number** CC filter: list up to six CC numbers and choose **Pass Listed** (only those survive) or **Block Listed** (everything but those). Non-CC messages always pass.

Keep just the mod wheel and CC 74 for a synth that misbehaves on stray controllers, or strip one noisy encoder without losing the rest. Set a slot to **Off** (128) to leave it unused.

#### PARAMETERS

| | |
|---|---|
| `Mode` | Pass Listed = only the listed CCs survive; Block Listed = they are removed. |
| `CC 1–6` | CC numbers 0–127; Off (128) = slot unused. |

◉ Every parameter accepts amber mod wires (MIDI In signals, LFO, Attractor, Life stats, etc.).



### Hold

`In: 1× MIDI · Out: 1× MIDI`

A **monophonic note latch**: each new note-on replaces the previous held note. The prior note-off is sent when the next note arrives — legato overlap on a single voice with no extra parameters.




### Chance

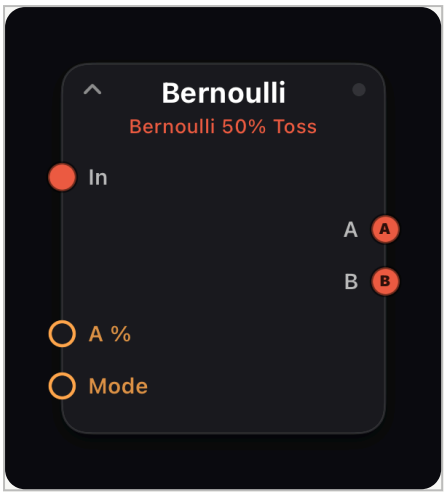
`In: 1× MIDI · Out: 1× MIDI`

Probabilistic gate: each note-on passes with **Chance** %; note-offs always pass so notes never stick. Thin dense streams or add organic drop-outs to arps and generators.

#### PARAMETERS

| | |
|---|---|
| `Chance %` | Probability 0–100% that an event fires. |

◉ Every parameter accepts amber mod wires (MIDI In signals, LFO, Attractor, Life stats, etc.).




### Bernoulli

`In: 1× MIDI · Out: A · B`

`Mod out: the last toss (1 = A, 0 = B)`

A **coin toss** for every note (after Mutable Instruments' Branches): heads goes out **A**, tails out **B**. Nothing is dropped — that's **Chance** — Bernoulli *routes*. Feed one hat pattern in and send **A** to the closed hat and **B** to the open one, or spread one arp across two instruments.

**Toss** flips a fresh coin for every note. **Toggle** flips it to decide whether to *switch* sides: a low **A %** holds long runs on one side, 100% alternates strictly.

Clock pulses are tossed too (on their own coin), so a **Clock** wired through drives two generators in complementary random patterns. A note-off always follows its note-on; CC, pitch bend and pressure reach both outputs.

#### OUTPUT PORTS

| | |
|---|---|
| A | Heads — **A %** of the notes. |
| B | Tails — the rest. |

#### PARAMETERS

| | |
|---|---|
| A % | Probability 0–100% that a note goes out **A** (default 50). |
| Mode | **Toss** — a coin per note; **Toggle** — the coin decides whether to switch sides. |

◉ Every parameter accepts amber mod wires (MIDI In signals, LFO, Attractor, Life stats, etc.).




### Divide

`In: 1× MIDI · Out: 1× MIDI`

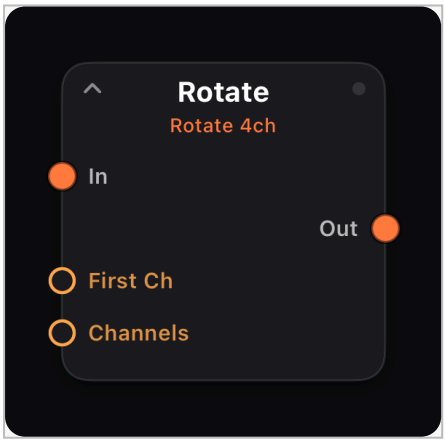
Passes every **Nth** note-on after skipping **Offset** hits — accent every third note in a run, or extract a sparse melody from busy generator output.

Also **divides clock pulses**: every **Nth** pulse from a **Clock** node passes through (separate counter from notes). Halve or quarter a master clock into a generator's **Clock** input.

PARAMETERS

| | |
|---|---|
| `Every N` | Pass every Nth note-on or clock pulse (1–8). |
| `Offset` | Skip this many events before counting (0–7). |

◉ Every parameter accepts amber mod wires (MIDI In signals, LFO, Attractor, Life stats, etc.).




### Rotate

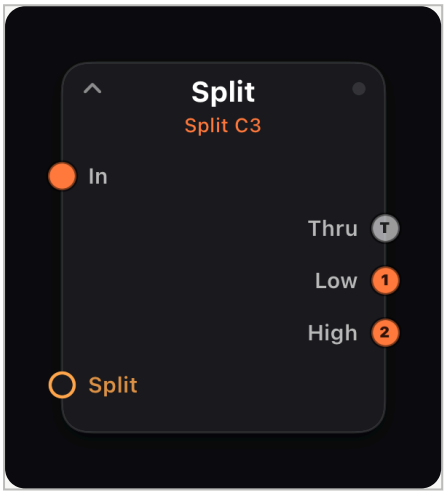
`In: 1× MIDI · Out: 1× MIDI`

**Round-robin channel assignment**: first note goes to **First Ch**, then increments, wrapping after **Channels** count. Spread one stream across multitimbral timbres or layered synth patches.

PARAMETERS

| | |
|---|---|
| `First Ch` | MIDI channel for the first note (1–16). |
| `Channels` | How many channels to cycle through (1–16). |

◉ Every parameter accepts amber mod wires (MIDI In signals, LFO, Attractor, Life stats, etc.).




### Split

`In: 1× MIDI · Out: T · Low · High`

Keyboard split at **Split** pitch. Notes at or below the split point route to **Low**; above to **High**. **T** outputs the full input.

CC, pitch bend and aftertouch copy to both zones.

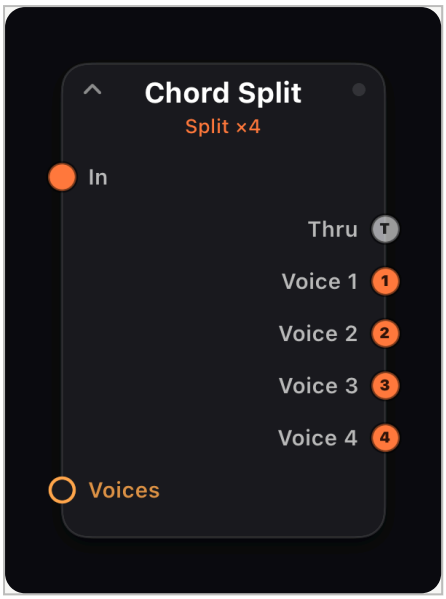
OUTPUT PORTS

| | |
|---|---|
| `T` | Thru — full unmodified input. |
| `Low` | Notes at or below **Split** pitch. |
| `High` | Notes above **Split** pitch. |

PARAMETERS

| | |
|---|---|
| `Split` | Split pitch 0–127 (default middle C = 60). |

◉ Every parameter accepts amber mod wires (MIDI In signals, LFO, Attractor, Life stats, etc.).




### Chord Split

`In: 1× MIDI · Out: T · 1…n voices`

Decomposes each chord into separate **voice** outputs (up to **Voices**). Lowest note → port 1, next → port 2, etc. **T** passes the untouched chord.

Route each voice to different processing chains for per-note effects.

#### OUTPUT PORTS

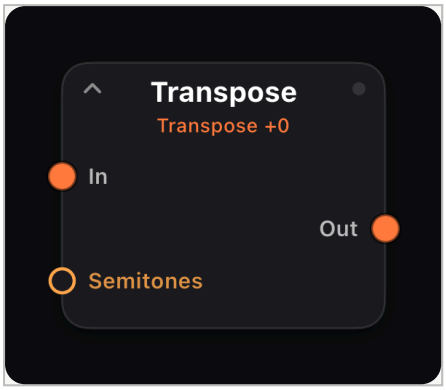
| | |
|---|---|
| T | Thru — full chord unchanged. |
| 1…n | Individual chord voices (count = **Voices**), lowest note on port 1. |

#### PARAMETERS

| | |
|---|---|
| `Voices` | Number of voice outputs 2–8. |

◉ Every parameter accepts amber mod wires (MIDI In signals, LFO, Attractor, Life stats, etc.).

# Notes — transform, chords, mono lines, echo, humanize




### Transpose

`In: 1× MIDI · Out: 1× MIDI`

Shifts every note by **Semitones** (−24…+24). Note-offs track the transposed pitch so voices don't stick. Instant key change or fine interval tweak (+7 = fifth up).

PARAMETERS

| | |
|---|---|
| `Semitones` | Interval shift −24 … +24 semitones. |

◉ Every parameter accepts amber mod wires (MIDI In signals, LFO, Attractor, Life stats, etc.).




### Mirror

`In: 1× MIDI · Out: 1× MIDI`

Inverts pitch around **Pivot** — notes above become equidistant below and vice versa. Set pivot to the tonic for inverted lines that stay in register.

PARAMETERS

| | |
|---|---|
| `Pivot` | Mirror axis 0–127 (MIDI note number). |

◉ Every parameter accepts amber mod wires (MIDI In signals, LFO, Attractor, Life stats, etc.).




### Chord

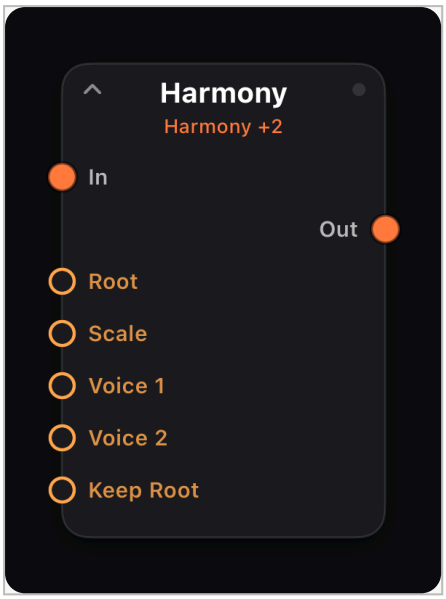
`In: 1× MIDI · Out: 1× MIDI`

Expands each note-on into a full **Chord** type (Major, Minor, 7ths, sus, etc.). **Spread** staggers voice onsets in milliseconds for a strummed feel. One-finger pad chords.

#### PARAMETERS

| | |
|---|---|
| `Chord` | Chord quality (Major, Minor, dim, aug, 7ths, sus...). |
| `Spread ms` | Delay between chord voices 0–100 ms. |

◉ Every parameter accepts amber mod wires (MIDI In signals, LFO, Attractor, Life stats, etc.).




### Harmony

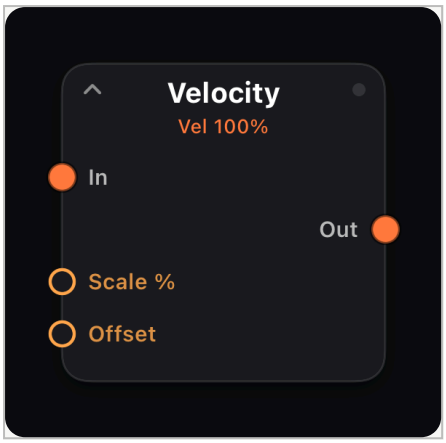
`In: 1× MIDI · Out: 1× MIDI`

Adds up to two **diatonic voices** at scale-degree offsets from **Root** + **Scale**. **Keep Root** On = original note plus harmonies; Off = harmony voices only.

#### PARAMETERS

| | |
|---|---|
| `Root` | Scale root (C, C#, D...). |
| `Scale` | Scale for voice selection — built-in or one of the patch's custom scales. |
| `Voice 1` | Scale-degree offset for first harmony (−7...+7). |
| `Voice 2` | Scale-degree offset for second harmony (−7...+7). |
| `Keep Root` | On = pass input note plus harmonies; Off = harmonies only. |

◉ Every parameter accepts amber mod wires (MIDI In signals, LFO, Attractor, Life stats, etc.).




### Velocity

`In: 1× MIDI · Out: 1× MIDI`

Scales note-on velocity: multiply by **Scale** % then add **Offset**. Compress or expand dynamics from a controller, sequencer or humanized stream.

PARAMETERS

| | |
|---|---|
| `Scale %` | Velocity multiplier 0–200%. |
| `Offset` | Added after scaling (−64…+64). |

◉ Every parameter accepts amber mod wires (MIDI In signals, LFO, Attractor, Life stats, etc.).




### Note Length

`In: 1× MIDI · Out: 1× MIDI`

Forces a fixed **Length** (clock division) on every note. Input note-offs are ignored; the node issues its own off at grid time — staccato lines from legato input.

PARAMETERS

| | |
|---|---|
| `Length` | Fixed note length as a clock division (`4/1` … `1/64`, dotted and triplet variants included). |

◉ Every parameter accepts amber mod wires (MIDI In signals, LFO, Attractor, Life stats, etc.).




### Mono

`In: 1× MIDI · Out: 1× MIDI`

A **voice limiter**: only one note sounds at a time. **Priority** picks the winner — Last (newest press steals), Low or High — and releasing the sounding key falls back to the best still-held key, like a classic mono synth.

**Mode** orders the switch: Retrig closes the old note before the new one; Legato opens the new note first, so a legato-mode synth slides instead of restarting its envelope. Put it before mono basses, leads, or **Glide**.

#### PARAMETERS

| | |
|---|---|
| `Priority` | Which held note wins the voice: Last, Low, or High. |
| `Mode` | Retrig = off-then-on; Legato = overlapped on-then-off. |

◉ Every parameter accepts amber mod wires (MIDI In signals, LFO, Attractor, Life stats, etc.).




### Glide

`In: 1× MIDI · Out: 1× MIDI`

**Pitch-bend portamento** (the 303 slide): a voice's first note passes through, and when its target note changes the change becomes a smooth pitch-bend ramp toward the new pitch over **Time** ms instead of a retrigger. Releasing back to a still-held key slides back down. **Polyphonic** — up to 8 voices glide independently: held notes are ranked by ascending pitch, each rank keeps its own voice, so whole chords slide when the harmony shifts.

Pitch bend is channel-wide, so each voice gets its own fixed output channel (**Channel Base** + voice index, wrapping mod 16) — the one-channel-per-voice trick MPE controllers use. **Bend Range** is announced on every voice channel as pitch-bend sensitivity (RPN 0,0), so the built-in **Analog** — and RPN-honoring hardware — lands exactly on pitch; jumps beyond the range re-anchor with a fresh note and keep sliding. **Always** mode glides even detached notes in from the previous pitch.

#### PARAMETERS

| | |
|---|---|
| `Time ms` | Glide time 5–2000 ms per slide. |
| `Bend Range` | Bend span in semitones (1–48); match the receiver. |
| `Mode` | Legato = only overlapping notes slide; Always = every note. |
| `Channel Base` | Lowest of the 8 per-voice output channels (1–16). |

◉ Every parameter accepts amber mod wires (MIDI In signals, LFO, Attractor, Life stats, etc.).




### Ratchet

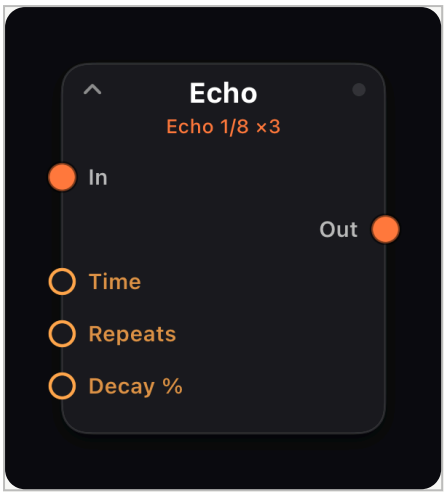
`In: 1× MIDI · Out: 1× MIDI`

On each incoming note-on, fires **Repeats** additional hits at **Rate** with **Decay** % velocity falloff and **Chance** % per hit. Machine-gun hi-hats and trap snares.

#### PARAMETERS

| | |
|---|---|
| `Repeats` | Extra hits per note-on 1–4. |
| `Rate` | Clock division `4/1` … `1/64`, each with a **dotted** (`1/8D`) and **triplet** (`1/8T`) variant. Follows host transport tempo. |
| `Decay %` | Velocity falloff per repeat 0–100%. |
| `Chance %` | Probability 0–100% that an event fires. |

◉ Every parameter accepts amber mod wires (MIDI In signals, LFO, Attractor, Life stats, etc.).




### Echo

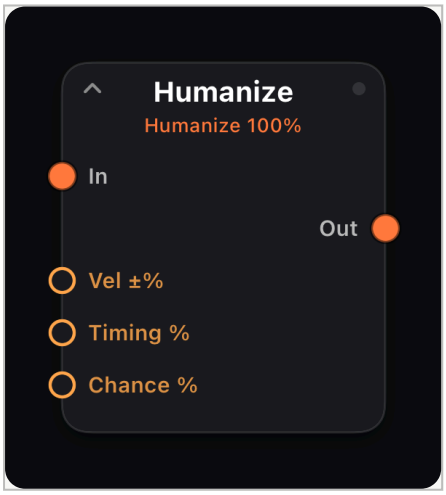
`In: 1× MIDI · Out: 1× MIDI`

Tempo-synced **MIDI delay**: **Repeats** echoes at **Time** division, each quieter by **Decay** %. Original notes pass through; echoes are added alongside.

#### PARAMETERS

| | |
|---|---|
| `Time` | Delay interval as a clock division (`4/1` … `1/64`, dotted and triplet variants included). |
| `Repeats` | Number of echo taps 1–8. |
| `Decay %` | Velocity reduction per repeat 0–100%. |

◉ Every parameter accepts amber mod wires (MIDI In signals, LFO, Attractor, Life stats, etc.).



### Humanize

`In: 1× MIDI · Out: 1× MIDI`

Adds controlled randomness: **Vel ±%** jitter, **Timing** % shift and **Chance** % dropouts. Place after sequencers for life; before **Snap** to tighten back up.

PARAMETERS

| | |
|---|---|
| `Vel ±%` | Random velocity variation 0–100%. |
| `Timing %` | Random timing jitter 0–100%. |
| `Chance %` | Probability 0–100% that an event fires. |

◉ Every parameter accepts amber mod wires (MIDI In signals, LFO, Attractor, Life stats, etc.).




### AU MIDI Effect

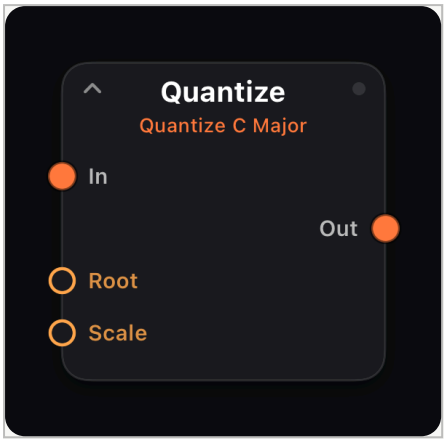
`In: 1× MIDI · Out: 1× MIDI`

A **hosted third-party AUv3 MIDI effect** — any MIDI-processor plugin installed on the device (arpeggiators, chord tools, generative sequencers, harmonisers) sits in the MIDI chain like a built-in node: the events routed into it go through the plugin and whatever it emits comes out the other side, at sample offsets. Pick the plugin in the inspector (**Choose…**) and open its own window from there; its settings are saved with the patch, and its parameters appear as node parameters you can bind and modulate. The plugin follows the transport — tempo, beat and play/stop — so arps and step plugins stay in time.

Clock pulses bypass the plugin, and a plugin that fails to render passes its input through rather than swallowing notes.

**Standalone app only.** Inside a host like AUM or Logic the node stays in the patch but passes MIDI through untouched.

# Pitch & Scale — quantize and arpeggiate




### Quantize

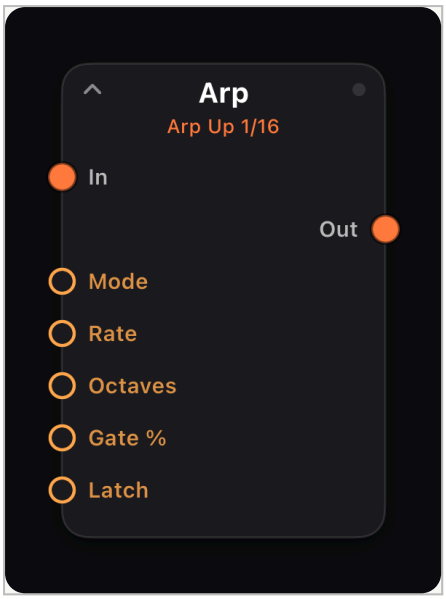
`In: 1× MIDI · Out: 1× MIDI`

Snaps every note to **Root** + **Scale**. Essential after **Random Notes**, **Walk**, or emergence generators; also sweetens live keyboard input into a chosen key.

PARAMETERS

| | |
|---|---|
| `Root` | Scale root (C, C#, D...). |
| `Scale` | Scale name (Major, Minor, Pentatonic, etc.). |

◉ Every parameter accepts amber mod wires (MIDI In signals, LFO, Attractor, Life stats, etc.).




### Arp

`In: 1× MIDI · Out: 1× MIDI`

Arpeggiates held notes in **Mode** (Up, Down, Up/Down, Random…) at **Rate** across **Octaves** with **Gate** % length. **Latch** keeps running after keys are released.

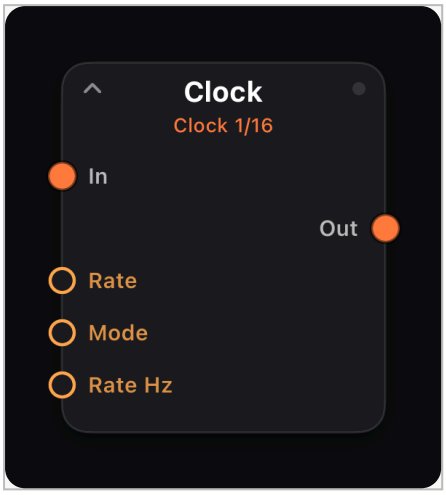
Hold a chord for running trance and synth-wave leads.

#### PARAMETERS

| | |
|---|---|
| `Mode` | Arp direction / pattern. |
| `Rate` | Clock division `4/1` … `1/64`, each with a **dotted** (`1/8D`) and **triplet** (`1/8T`) variant. Follows host transport tempo. |
| `Octaves` | Range extension 0–4 octaves. |
| `Gate %` | Note length as percent of one grid step (1–100). |
| `Latch` | Off = stop when keys released; On = keep arpeggiating. |

◉ Every parameter accepts amber mod wires (MIDI In signals, LFO, Attractor, Life stats, etc.).

# Generate & Modulate — clock, euclidean, LFOs, CC tools




### Clock

`In: 1× MIDI (pass-through) · Out: 1× MIDI + clock pulses`

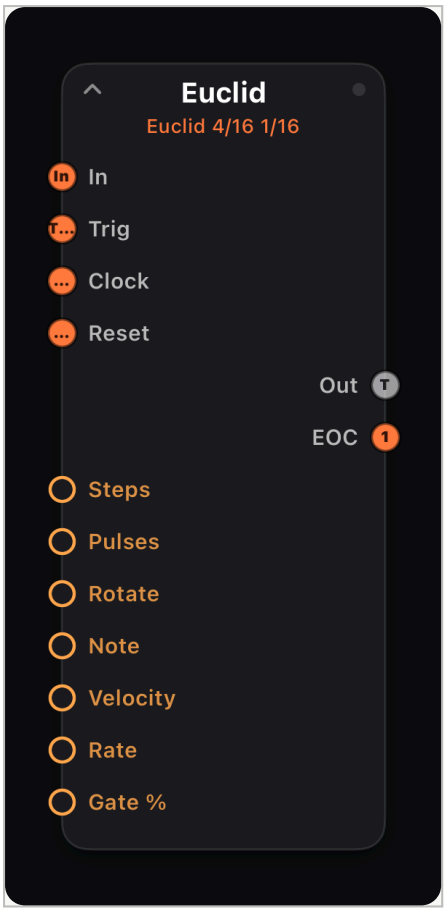
A dedicated **timing pulse** generator. Emits one clock pulse per **Rate** step on the MIDI stream (in addition to passing input through). Wire **Out** to the **Clock** input of sequencers and generators to drive many nodes from one master tempo.

**Sync** steps on a clock division, locked to the host transport (pauses when stopped). **Free** ticks at a **Rate Hz** you set — independent of the transport *and* the tempo, so it keeps going while stopped and doesn't speed up with the song.

#### PARAMETERS

| | |
|---|---|
| `Rate` | Clock division `4/1` … `1/64`, each with a **dotted** (`1/8D`) and **triplet** (`1/8T`) variant. Follows host transport tempo. |
| `Mode` | Sync = follow host transport on the clock division; Free = run at Rate Hz. |
| `Rate Hz` | Free mode only: cycles per second, 0.05–20 Hz. Shown in place of the clock division once Mode is Free; ignored in Sync. |

◉ Every parameter accepts amber mod wires (MIDI In signals, LFO, Attractor, Life stats, etc.).




### Euclid

`In · Trig · Clock · Reset · Out: 1× MIDI + EOC`

**Euclidean rhythm** generator: distributes **Pulses** hits evenly across **Steps**, with **Rotate** phase offset. Fires **Note** at **Rate** or external clock.

Input on **In** passes through — layer polyrhythms on top of existing material without replacing it.

#### INPUT PORTS

| | |
|---|---|
| `In` | Primary MIDI path — incoming events pass through unchanged to Out. |
| `Trig` | External step clock: each note-on advances the internal pattern one step (overrides **Rate**). |
| `Clock` | Wire from a **Clock** node — one pulse per step (mutually exclusive with **Trig**). |
| `Reset` | A note-on or clock pulse restarts the pattern at step 1 (on either clock mode) — wire a master lane's EOC or a performance pad here to re-align lanes. |

#### OUTPUT PORTS

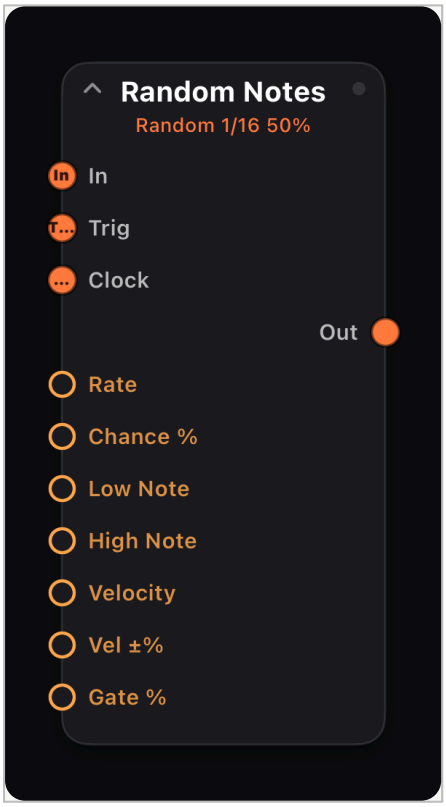
| | |
|---|---|
| `Out` | Euclidean note hits; input passes through alongside. |

| | |
|---|---|
| `EOC` | End-of-cycle trigger — one clock pulse each time the pattern wraps to step 1. Wire into another node's **Trig** or **Clock** to chain patterns. |

### PARAMETERS

| | |
|---|---|
| `Steps` | Pattern length 1–16. |
| `Pulses` | Number of hits in the pattern 0–16. |
| `Rotate` | Rotate pattern start 0–15. |
| `Note` | MIDI note number 0–127. |
| `Velocity` | Note-on velocity 1–127. |
| `Rate` | Clock division `4/1` … `1/64`, each with a **dotted** (`1/8D`) and **triplet** (`1/8T`) variant. Follows host transport tempo. |
| `Gate %` | Note length as percent of one grid step (1–100). |

◉ Every parameter accepts amber mod wires (MIDI In signals, LFO, Attractor, Life stats, etc.).



### Random Notes

`In · Trig · Clock · Out: 1× MIDI`

Probabilistic melody generator at **Rate**: random pitches between **Low/High Note** with **Chance** % and **Vel ±%** variation.

Follow with **Quantize** unless you want chromatic chaos.

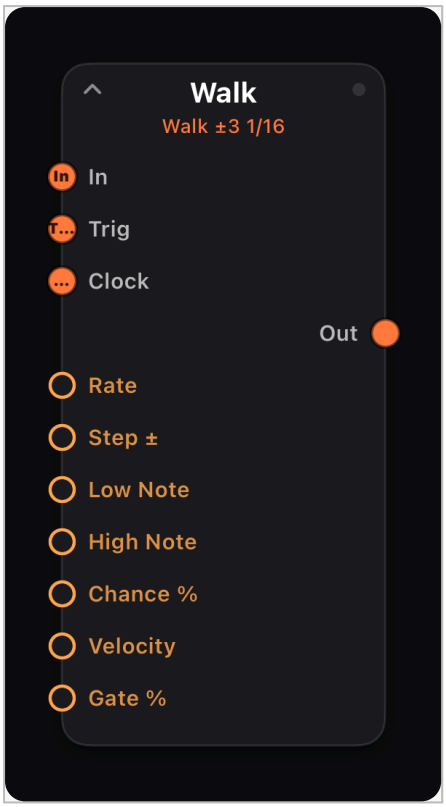
#### INPUT PORTS

| | |
|---|---|
| `In` | Primary MIDI path — incoming events pass through unchanged to Out. |
| `Trig` | External step clock: each note-on advances the internal pattern one step (overrides **Rate**). |
| `Clock` | Wire from a **Clock** node — one pulse per step (mutually exclusive with **Trig**). |

#### PARAMETERS

| | |
|---|---|
| `Rate` | Clock division `4/1` … `1/64`, each with a **dotted** (`1/8D`) and **triplet** (`1/8T`) variant. Follows host transport tempo. |
| `Chance %` | Probability 0–100% that an event fires. |
| `Low Note` | Lowest pitch in range (0–127). |
| `High Note` | Highest pitch in range (0–127). |
| `Velocity` | Note-on velocity 1–127. |
| `Vel ±%` | Random velocity spread 0–100%. |
| `Gate %` | Note length as percent of one grid step (1–100). |

◉ Every parameter accepts amber mod wires (MIDI In signals, LFO, Attractor, Life stats, etc.).




### Walk

`In · Trig · Clock · Out: 1× MIDI`

**Random-walk melody**: each tick moves ±**Step** semitones, clamped to **Low/High Note**. Wandering, self-similar lines for ambient and generative bass.

#### INPUT PORTS

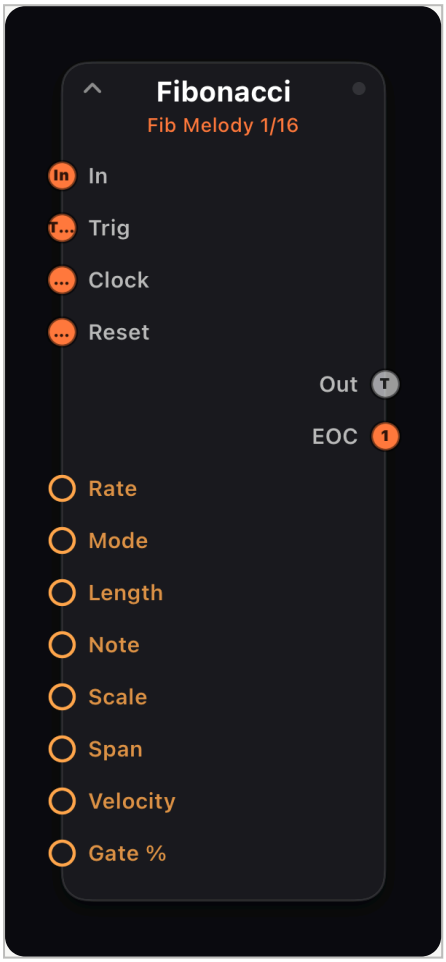
| | |
|---|---|
| `In` | Primary MIDI path — incoming events pass through unchanged to Out. |
| `Trig` | External step clock: each note-on advances the internal pattern one step (overrides **Rate**). |
| `Clock` | Wire from a **Clock** node — one pulse per step (mutually exclusive with **Trig**). |

#### PARAMETERS

| | |
|---|---|
| `Rate` | Clock division `4/1` … `1/64`, each with a **dotted** (`1/8D`) and **triplet** (`1/8T`) variant. Follows host transport tempo. |
| `Step ±` | Max semitone jump per tick 1–12. |
| `Low Note` | Lowest pitch in range (0–127). |
| `High Note` | Highest pitch in range (0–127). |

| | |
|---|---|
| `Chance %` | Probability 0–100% that an event fires. |
| `Velocity` | Note-on velocity 1–127. |
| `Gate %` | Note length as percent of one grid step (1–100). |

◉ Every parameter accepts amber mod wires (MIDI In signals, LFO, Attractor, Life stats, etc.).




### Fibonacci

`In · Trig · Clock · Reset · Out: 1× MIDI + EOC`

Two modes: **Melody** walks Fibonacci scale degrees; **Rhythm** fires only on Fibonacci step indices (1, 2, 3, 5, 8…). Self-similar phrases that never quite repeat.

#### INPUT PORTS

| | |
|---|---|
| `In` | Primary MIDI path — incoming events pass through unchanged to Out. |

| | |
|---|---|
| `Trig` | External step clock: each note-on advances the internal pattern one step (overrides **Rate**). |
| `Clock` | Wire from a **Clock** node — one pulse per step (mutually exclusive with **Trig**). |
| `Reset` | A note-on or clock pulse restarts the pattern at step 1 (on either clock mode) — wire a master lane's EOC or a performance pad here to re-align lanes. |

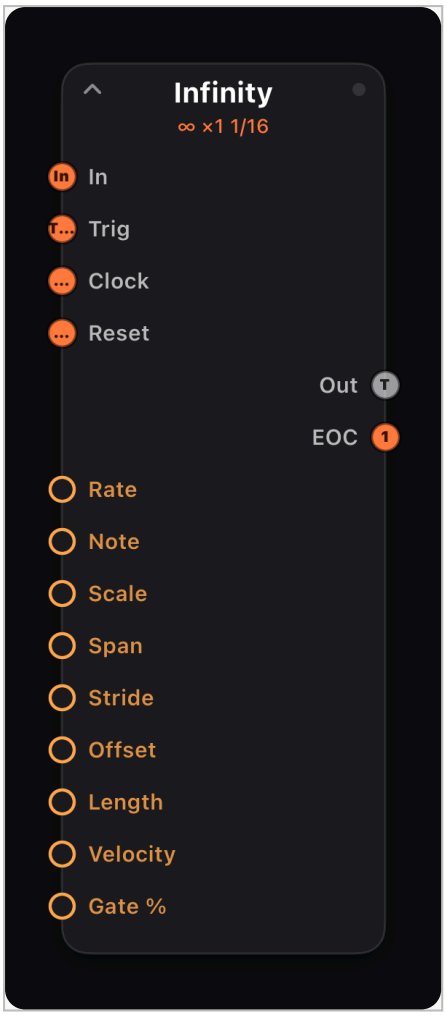
### OUTPUT PORTS

| | |
|---|---|
| `Out` | Generated notes; input passes through alongside. |
| `EOC` | End-of-cycle trigger — one clock pulse each time the pattern wraps to step 1. Wire into another node's **Trig** or **Clock** to chain patterns. |

### PARAMETERS

| | |
|---|---|
| `Rate` | Clock division `4/1` … `1/64`, each with a **dotted** (`1/8D`) and **triplet** (`1/8T`) variant. Follows host transport tempo. |
| `Mode` | Melody = pitch walk; Rhythm = Fibonacci hit pattern. |
| `Length` | Pattern length 2–16. |
| `Note` | MIDI note number 0–127. |
| `Scale` | Scale for melody mode. |
| `Span` | Scale-degree span 2–24. |
| `Velocity` | Note-on velocity 1–127. |
| `Gate %` | Note length as percent of one grid step (1–100). |

◉ Every parameter accepts amber mod wires (MIDI In signals, LFO, Attractor, Life stats, etc.).




### Infinity

`In · Trig · Clock · Reset · Out: 1× MIDI + EOC`

Per Nørgård's **infinity series** — a fractal melody played as scale degrees around **Note**. The sequence contains itself at every scale: **Stride** plays every Nth term, so 4, 16 or 64 reproduce the very same melody and 2 or 8 play its mirror inversion — a zoom knob for self-similarity you can hear.

**Offset** starts the loop deeper in the series, **Length** sets how many notes before the loop wraps (EOC pulses there), and **Span** caps how far the melody strays. Always lands in the chosen scale — no quantizer needed.

#### INPUT PORTS

| | |
|---|---|
| `In` | Primary MIDI path — incoming events pass through unchanged to Out. |
| `Trig` | External step clock: each note-on advances the internal pattern one step (overrides **Rate**). |

| | |
|---|---|
| `Clock` | Wire from a **Clock** node — one pulse per step (mutually exclusive with **Trig**). |
| `Reset` | A note-on or clock pulse restarts the pattern at step 1 (on either clock mode) — wire a master lane's EOC or a performance pad here to re-align lanes. |

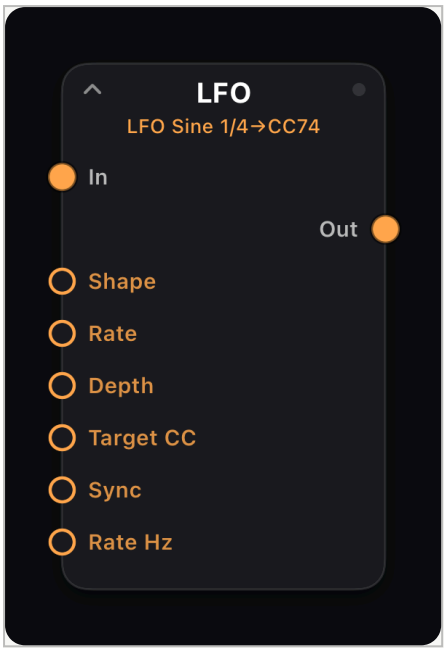
### OUTPUT PORTS

| | |
|---|---|
| `Out` | Generated notes; input passes through alongside. |
| `EOC` | End-of-cycle trigger — one clock pulse each time the pattern wraps to step 1. Wire into another node's **Trig** or **Clock** to chain patterns. |

### PARAMETERS

| | |
|---|---|
| `Rate` | Clock division `4/1` … `1/64`, each with a **dotted** (`1/8D`) and **triplet** (`1/8T`) variant. Follows host transport tempo. |
| `Note` | Centre note the degrees walk around. |
| `Scale` | Scale the degrees are drawn from. |
| `Span` | Maximum scale-degree excursion 2–24. |
| `Stride` | Play every Nth term (1–32) — 4/16/64 replay the series, 2/8 invert it. |
| `Offset` | Where in the series the loop starts (0–127). |
| `Length` | Loop length 2–64; the EOC pulse fires at the wrap. |
| `Velocity` | Note-on velocity 1–127. |
| `Gate %` | Note length as percent of one grid step (1–100). |

◉ Every parameter accepts amber mod wires (MIDI In signals, LFO, Attractor, Life stats, etc.).




### LFO

`In: — · Out: CC on Target CC`

**`Mod out:`** `main LFO signal (0–1)`

Low-frequency oscillator writing **Target CC** continuously. **Shape** (sine, triangle, square…), **Rate**, **Depth** and **Sync/Free** mode. In **Sync** the cycle is a clock division locked to the host transport; in **Free** it runs at **Rate Hz** — unaffected by transport or tempo.

The amber **mod out** port drives any parameter directly — wire to Echo **Decay**, filter cutoff, or another node's knob.

#### PARAMETERS

| | |
|---|---|
| `Shape` | Waveform (Sine, Triangle, Square, Saw, Random…). |
| `Rate` | Clock division `4/1` … `1/64`, each with a **dotted** (`1/8D`) and **triplet** (`1/8T`) variant. Follows host transport tempo. |
| `Depth` | Modulation amount 0–127. |
| `Target CC` | MIDI CC number to write (0–127). |
| `Sync` | Sync = tempo-locked clock division; Free = Hz-based rate. |
| `Rate Hz` | Free mode only: cycles per second, 0.05–20 Hz. Shown in place of the clock division once Mode is Free; ignored in Sync. |

◉ Every parameter accepts amber mod wires (MIDI In signals, LFO, Attractor, Life stats, etc.).




### CC Map

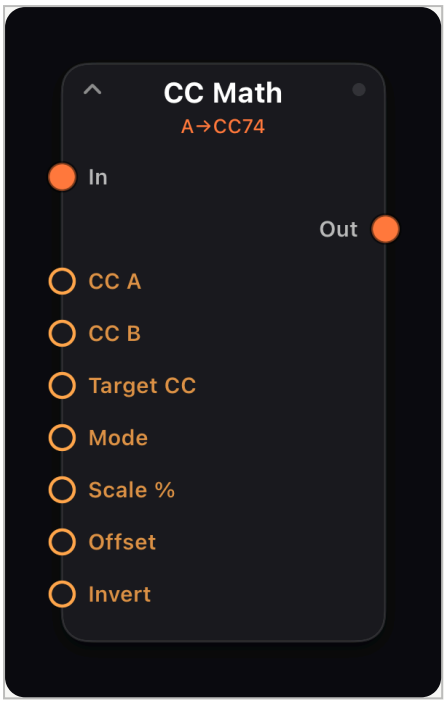
`In: 1× MIDI · Out: 1× MIDI`

Remaps **From CC** → **To CC** on **Channel** (0 = any). Duplicate a mod wheel to multiple targets or translate controller numbers for incompatible synths.

#### PARAMETERS

| | |
|---|---|
| `From CC` | Source CC number 0–127. |
| `To CC` | Destination CC number 0–127. |
| `Channel` | 0 = any channel; 1–16 = filter to channel. |

◉ Every parameter accepts amber mod wires (MIDI In signals, LFO, Attractor, Life stats, etc.).




### CC Math

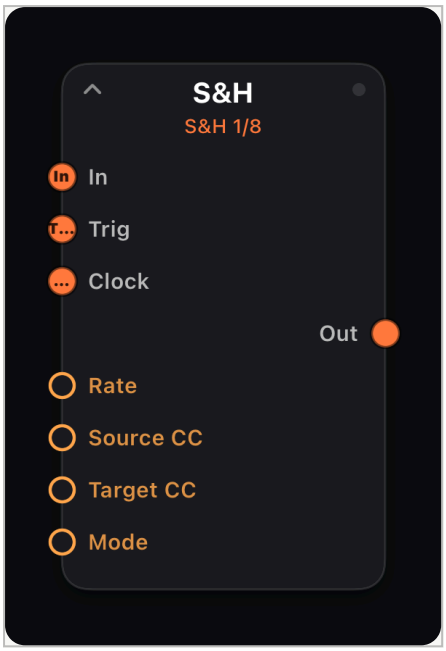
`In: 1× MIDI · Out: 1× MIDI + Target CC`

Combines incoming **CC A** and **CC B** values: A, A+B, A−B, Average, Max or Min — then **Scale**, **Offset** and optional **Invert** on **Target CC**.

#### PARAMETERS

| | |
|---|---|
| `CC A` | First operand CC 0–127. |
| `CC B` | Second operand CC 0–127. |
| `Target CC` | MIDI CC number to write (0–127). |
| `Mode` | Math operation applied to A and B. |
| `Scale %` | Output multiplier 0–200%. |
| `Offset` | Added after scaling −127…+127. |
| `Invert` | Off / On — flip output (127 − value). |

◉ Every parameter accepts amber mod wires (MIDI In signals, LFO, Attractor, Life stats, etc.).




### S&H

`In · Trig · Clock · Out: 1× MIDI`

Each **Rate** tick, **Sample** holds the current **Source CC** value or **Random** throws a new value on **Target CC**. Classic stepped and random filter sweeps.

#### INPUT PORTS

| | |
|---|---|
| `In` | Primary MIDI path — incoming events pass through unchanged to Out. |
| `Trig` | External step clock: each note-on advances the internal pattern one step (overrides **Rate**). |
| `Clock` | Wire from a **Clock** node — one pulse per step (mutually exclusive with **Trig**). |

#### PARAMETERS

| | |
|---|---|
| `Rate` | Clock division `4/1` … `1/64`, each with a **dotted** (`1/8D`) and **triplet** (`1/8T`) variant. Follows host transport tempo. |
| `Source CC` | CC to sample 0–127. |
| `Target CC` | MIDI CC number to write (0–127). |
| `Mode` | Sample = track input CC; Random = new value each step. |

◉ Every parameter accepts amber mod wires (MIDI In signals, LFO, Attractor, Life stats, etc.).




### Counter

`In: 1× MIDI · Out: 1× MIDI`

**`Mod out:`** `stepped count (0–1)`

A **stepped counter**: its value rises by one on every input event — a **note-on** or a **clock / EOC** pulse — and is exposed on the amber **mod out**. Wire it onto another node's indexed parameter (waveform, scale, clock division, **Selector**…) to walk through the options; set **Count** to the number of options so each input advances exactly one.

**Mode** picks Wrap (cycle), Up-Down (bounce) or Clamp (rise once and hold). Input passes through unchanged — the count never appears as MIDI.

#### PARAMETERS

| | |
|---|---|
| `Count` | Number of steps before wrapping / bouncing 2–32. |
| `Mode` | Wrap = cycle; Up-Down = bounce; Clamp = rise once and hold. |

◉ Every parameter accepts amber mod wires (MIDI In signals, LFO, Attractor, Life stats, etc.).




### Note→CC

`In: 1× MIDI · Out: 1× MIDI + Target CC`

Maps **Note #** or **Velocity** to **Target CC** on every note-on — keytracking and performance-driven automation without a separate LFO.

PARAMETERS

| | |
|---|---|
| `Source` | Note # = pitch drives CC; Velocity = dynamics drive CC. |
| `Target CC` | MIDI CC number to write (0–127). |

◉ Every parameter accepts amber mod wires (MIDI In signals, LFO, Attractor, Life stats, etc.).




### Smooth

`In: 1× MIDI · Out: 1× MIDI`

Slew-limits CC changes over **Time** ms. **CC** 0 smooths all CCs; otherwise only the named CC. Glides stepped sequencers and S&H into usable sweeps.

PARAMETERS

| | |
|---|---|
| `CC (0=all)` | Which CC to smooth (0 = all CC messages). |
| `Time ms` | Slew time 10–2000 ms. |

Every parameter accepts amber mod wires (MIDI In signals, LFO, Attractor, Life stats, etc.).




### Envelope

`In: 1× MIDI · Out: 1× MIDI + Target CC`

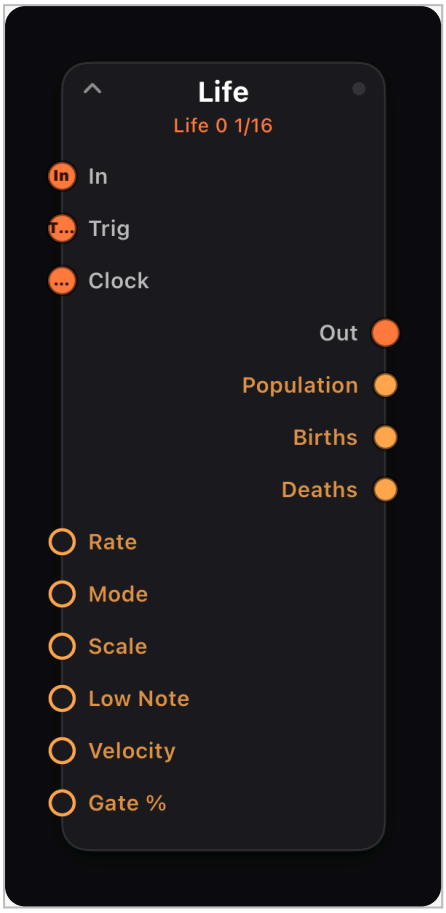
Per-note **Attack / Decay / Peak** envelope on **Target CC**, retriggered each note-on. Auto-filter sweeps and pluck brightness from your playing.

PARAMETERS

| | |
|---|---|
| `Target CC` | MIDI CC number to write (0–127). |
| `Attack ms` | Rise time 1–2000 ms. |
| `Decay ms` | Fall time 10–5000 ms. |
| `Peak` | Maximum CC value 1–127. |

Every parameter accepts amber mod wires (MIDI In signals, LFO, Attractor, Life stats, etc.).

# Emergence — generative, evolving patterns




### Life

`In · Trig · Clock · Out: 1× MIDI`

`Mod out: Population · Births · Deaths`

Conway's **Game of Life** on a 16×12 grid. **Pulse** mode fires notes from live-cell events; **Scan** reads columns as time and rows as scale degrees for looping ostinatos.

Draw a seed on the surface Life grid. Pair with **Quantize** or use built-in **Scale** mapping.

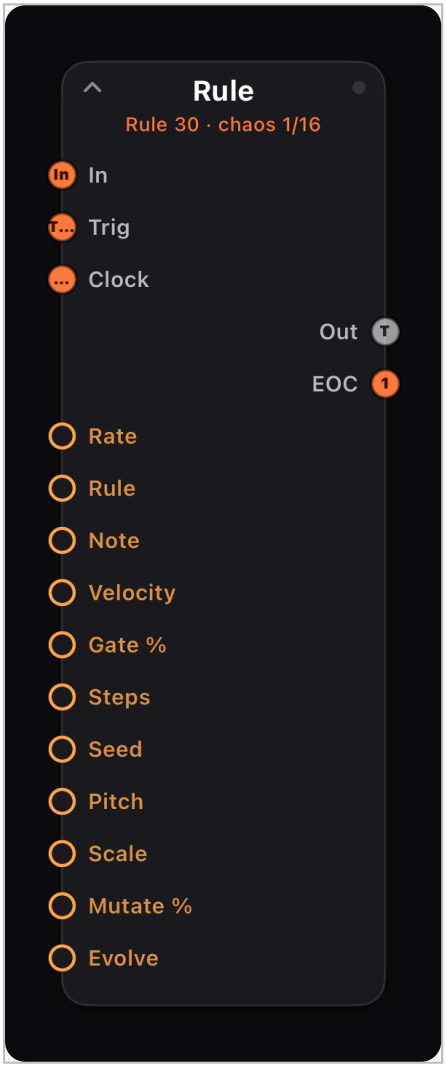
#### INPUT PORTS

| | |
|---|---|
| `In` | Primary MIDI path — incoming events pass through unchanged to Out. |
| `Trig` | External step clock: each note-on advances the internal pattern one step (overrides **Rate**). |
| `Clock` | Wire from a **Clock** node — one pulse per step (mutually exclusive with **Trig**). |

#### PARAMETERS

| | |
|---|---|
| `Rate` | Clock division `4/1` … `1/64`, each with a **dotted** (`1/8D`) and **triplet** (`1/8T`) variant. Follows host transport tempo. |
| `Mode` | Pulse = event-driven; Scan = column scanner. |
| `Scale` | Pitch mapping for live cells. |
| `Low Note` | Lowest pitch in range (0–127). |
| `Velocity` | Note-on velocity 1–127. |
| `Gate %` | Note length as percent of one grid step (1–100). |
| `Cell 1–192` | Seed pattern on the 16×12 Life grid (bottom row = lowest pitch). Surface editor. |

◉ Every parameter accepts amber mod wires (MIDI In signals, LFO, Attractor, Life stats, etc.).



### Rule

`In · Trig · Clock · Out: 1× MIDI + EOC`

1D **Wolfram cellular automaton** (**Rule** 0–255; notable rules show tags like **90 · fractal**). **Trigger** pitch mode fires from live cells; **Melody** maps columns to scale degrees.

**Seed** Center or Random initializes the row. **Mutate %** flips random cells each generation (0% = pure automaton). Dead or frozen rows **reseed** automatically so every rule stays musical.

**Evolve** morphs the row every 1/2/4/8 steps (**Loop** = once per pattern wrap, the default). **Register** view edits live cells (same idea as the Life grid seed editor).

#### INPUT PORTS

| | |
|---|---|
| `In` | Primary MIDI path — incoming events pass through unchanged to Out. |
| `Trig` | External step clock: each note-on advances the internal pattern one step (overrides **Rate**). |
| `Clock` | Wire from a **Clock** node — one pulse per step (mutually exclusive with **Trig**). |

#### OUTPUT PORTS

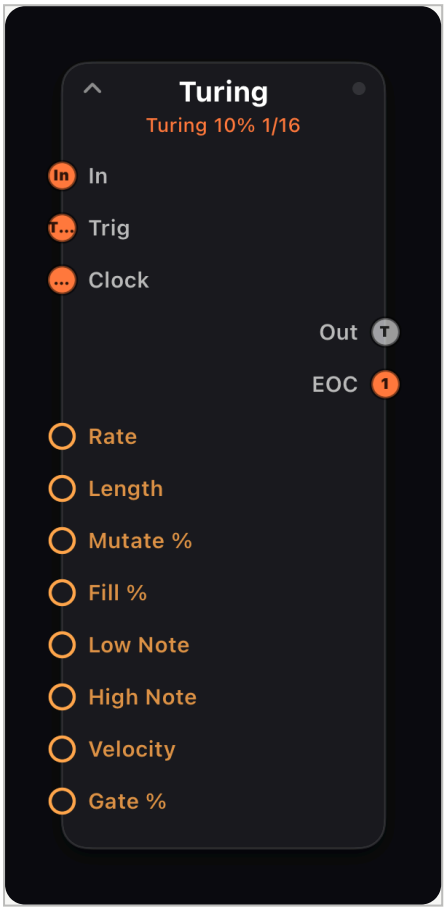
| | |
|---|---|
| `Out` | Cellular-automaton notes; input passes through alongside. |
| `EOC` | End-of-cycle trigger — one clock pulse each time the pattern wraps to step 1. Wire into another node's **Trig** or **Clock** to chain patterns. |

#### PARAMETERS

| | |
|---|---|
| `Rate` | Clock division `4/1` … `1/64`, each with a **dotted** (`1/8D`) and **triplet** (`1/8T`) variant. Follows host transport tempo. |
| `Rule` | Wolfram rule number 0–255 (notable rules tagged in the UI). |
| `Note` | MIDI note number 0–127. |
| `Velocity` | Note-on velocity 1–127. |
| `Gate %` | Note length as percent of one grid step (1–100). |
| `Steps` | Pattern length 1–16. |
| `Seed` | Center = single cell; Random = scattered start. |
| `Pitch` | Trigger = rhythm from cells; Melody = pitched output. |
| `Scale` | Scale for Melody mode. |

| | |
|---|---|
| `Mutate %` | Random cell flips per generation (default 10%). |
| `Evolve` | Loop, 1 Step, 2 Steps, 4 Steps or 8 Steps between row morphs. |
| `Cell 1–16` | Live row bits — tap or drag in the **Register** view to poke cells on/off. |

◉ Every parameter accepts amber mod wires (MIDI In signals, LFO, Attractor, Life stats, etc.).




### Turing

`In · Trig · Clock · Out: 1× MIDI + EOC`

Self-mutating **shift register**: **Length** bits, **Fill** % density, **Mutate** % bit-flip chance per step. Pattern slowly evolves from locked to chaotic.

#### INPUT PORTS

| | |
|---|---|
| `In` | Primary MIDI path — incoming events pass through unchanged to Out. |
| `Trig` | External step clock: each note-on advances the internal pattern one step (overrides **Rate**). |

| | |
|---|---|
| `Clock` | Wire from a **Clock** node — one pulse per step (mutually exclusive with **Trig**). |

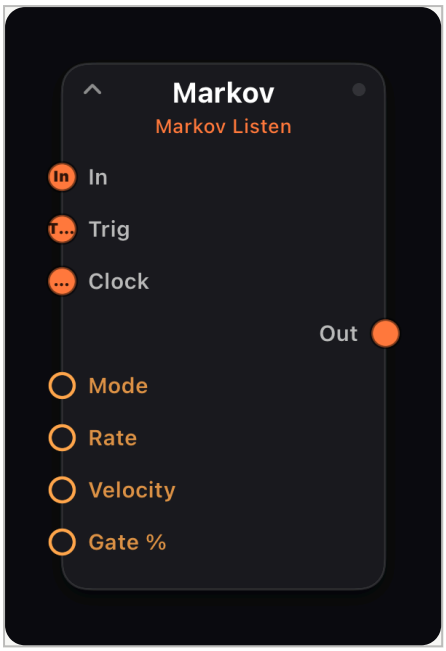
### OUTPUT PORTS

| | |
|---|---|
| `Out` | Shift-register notes; input passes through alongside. |
| `EOC` | End-of-cycle trigger — one clock pulse each time the pattern wraps to step 1. Wire into another node's **Trig** or **Clock** to chain patterns. |

### PARAMETERS

| | |
|---|---|
| `Rate` | Clock division `4/1` … `1/64`, each with a **dotted** (`1/8D`) and **triplet** (`1/8T`) variant. Follows host transport tempo. |
| `Length` | Register length 2–16. |
| `Mutate %` | Bit-flip probability per step 0–100%. |
| `Fill %` | Initial ones density 1–100%. |
| `Low Note` | Lowest pitch in range (0–127). |
| `High Note` | Highest pitch in range (0–127). |
| `Velocity` | Note-on velocity 1–127. |
| `Gate %` | Note length as percent of one grid step (1–100). |

◉ Every parameter accepts amber mod wires (MIDI In signals, LFO, Attractor, Life stats, etc.).




### Markov

`In · Trig · Clock · Out: 1× MIDI`

Learns note transitions from incoming MIDI. **Listen** = pass-through training; **Generate** improvises at **Rate** in the learned style.

Feed it a melody loop in Listen mode, then switch to Generate for variations.

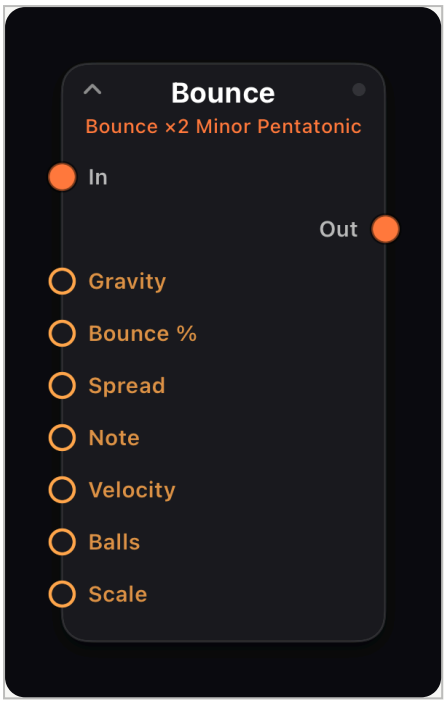
#### INPUT PORTS

| | |
|---|---|
| `In` | Primary MIDI path — incoming events pass through unchanged to Out. |
| `Trig` | External step clock: each note-on advances the internal pattern one step (overrides **Rate**). |
| `Clock` | Wire from a **Clock** node — one pulse per step (mutually exclusive with **Trig**). |

#### PARAMETERS

| | |
|---|---|
| `Mode` | Listen = learn + pass through; Generate = improvise. |
| `Rate` | Clock division `4/1` … `1/64`, each with a **dotted** (`1/8D`) and **triplet** (`1/8T`) variant. Follows host transport tempo. |
| `Velocity` | Note-on velocity 1–127. |
| `Gate %` | Note length as percent of one grid step (1–100). |

◉ Every parameter accepts amber mod wires (MIDI In signals, LFO, Attractor, Life stats, etc.).




### Bounce

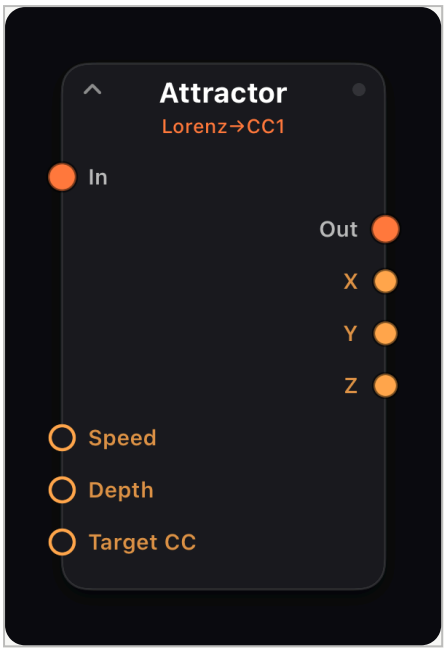
`In: 1× MIDI (pass-through) · Out: 1× MIDI`

2-D physics simulation: balls under **Gravity** bounce off walls and play scale degrees on impact. Organic polyrhythms from motion rather than fixed grids.

#### PARAMETERS

| | |
|---|---|
| `Gravity` | Downward acceleration 1–100. |
| `Bounce %` | Wall rebound energy 50–95%. |
| `Spread` | Initial horizontal spread 0–100%. |
| `Note` | MIDI note number 0–127. |
| `Velocity` | Note-on velocity 1–127. |
| `Balls` | Number of balls 1–4. |
| `Scale` | Pitch map for wall impacts. |

◉ Every parameter accepts amber mod wires (MIDI In signals, LFO, Attractor, Life stats, etc.).




### Attractor

`In: — · Out: CC on Target CC`

**`Mod out:`** `X · Y · Z`

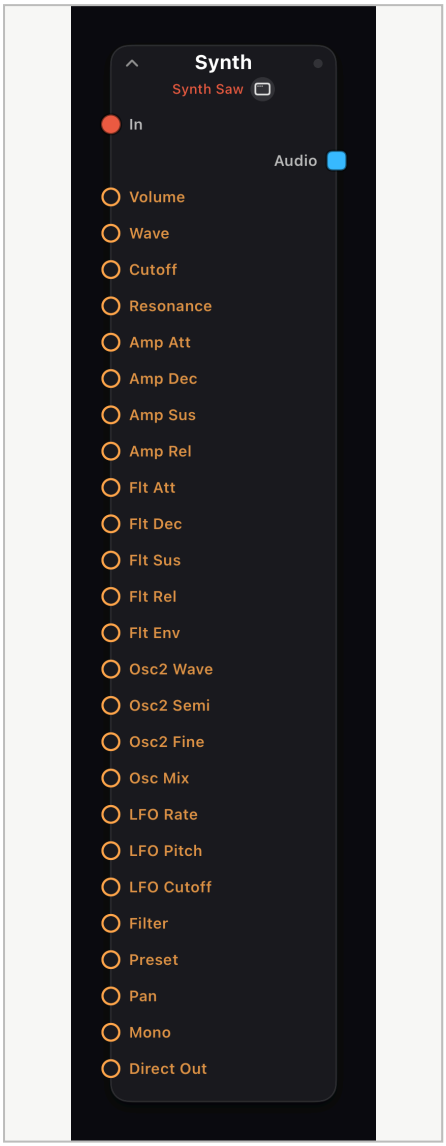
Lorenz **chaotic attractor** driving **Target CC**. Never-repeating organic modulation — wire **X**, **Y**, **Z** mod ports to cutoff, resonance and LFO rate for evolving textures.

#### PARAMETERS

| | |
|---|---|
| `Speed` | Evolution rate 1–100. |
| `Depth` | Output amplitude 0–127. |
| `Target CC` | MIDI CC number to write (0–127). |

◉ Every parameter accepts amber mod wires (MIDI In signals, LFO, Attractor, Life stats, etc.).

# **Instruments** — built-in synth and drum kit, and gear on a MIDI output




## Analog

`In: 1× MIDI · Out: — (audio sink)`

Terminal **polyphonic subtractive synth**. MIDI on **In** is rendered to audio inside Konstrukt — nothing passes downstream. Two oscillators (the second tuned in semitones and cents against the first, blended by **Osc Mix**), a resonant lowpass — a 24 dB transistor **Ladder** or the 12 dB **SVF** — dual ADSR envelopes (amplitude + filter) and an LFO that can add vibrato and wobble the cutoff — the same LFO the mod wheel drives. Under them: a **Sub** oscillator and **Noise**, pulse width with LFO PWM, **Drive** into the filter, key and velocity tracking, a **Pitch Env** for drops

and blips, and **Poly / Mono / Legato** voicing with the Analog's own **Glide**. **Width** spreads the voices across the stereo field.

Wire sequencers, arps or keyboards into one or more **Analog** nodes for a self-contained patch. Multiple instruments sum at the output.

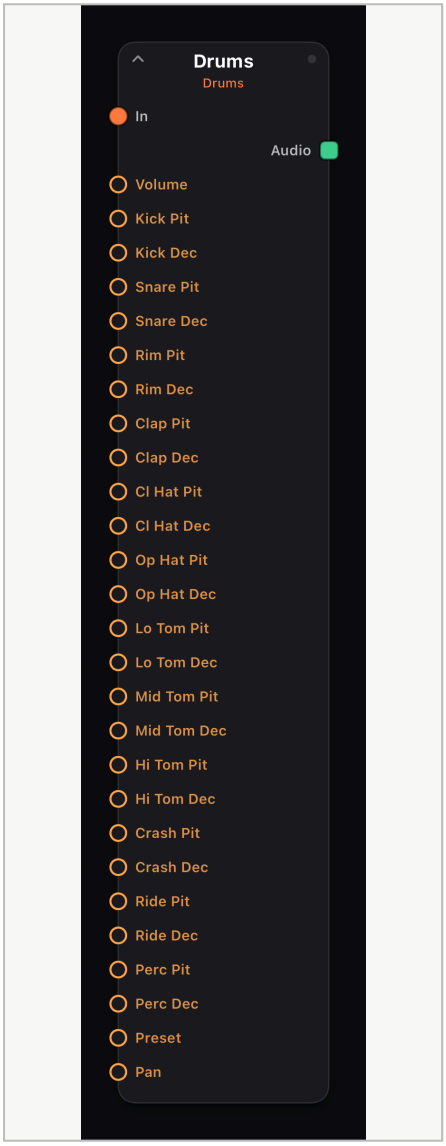
### PARAMETERS

| | |
|---|---|
| `Preset` | Sound preset: 57 sounds on seven shelves — Bass (Sub Bass, Acid Bass, Reese, 808 Bass, Moog Bass, Picked Bass ...), Lead (Bright Lead, Hoover, Mono Lead, Chip Lead, Theremin ...), Keys (Soft Keys, Organ, Clavinet, Wurli, Tine Keys ...), Pads (Warm Pad, Juno Pad, PWM Strings, Analog Strings, Shimmer Pad ...), Plucks (Marimba, Kalimba, Guitar Pluck, Steel Drum ...), Chords (Deep Stab, Rave Stab, Garage Chord, Skank) and FX & Perc (Synth Kick, Zap, Riser, Noise Sweep, Synth Tom ...). Picking one fills the controls below so you can tweak from there. |
| `Volume` | Master level 0–1. |
| `Wave` | Oscillator 1 waveform: Sine, Tri, Saw, Square. |
| `Cutoff` | Lowpass cutoff 0–100. |
| `Resonance` | Filter resonance 0–100 (the Ladder self-oscillates near the top). |
| `Filter` | Ladder (default): a 24 dB transistor ladder — fat, warm, bass thins as Resonance rises. SVF: the gentler 12 dB state-variable. Patches saved before the switch keep the SVF. |
| `Amp Att` | Amplitude attack 0–2000 ms. |
| `Amp Dec` | Amplitude decay 0–5000 ms. |
| `Amp Sus` | Amplitude sustain 0–100%. |
| `Amp Rel` | Amplitude release 0–5000 ms. |
| `Flt Att` | Filter envelope attack 0–2000 ms. |
| `Flt Dec` | Filter envelope decay 0–5000 ms. |
| `Flt Sus` | Filter envelope sustain 0–100%. |
| `Flt Rel` | Filter envelope release 0–5000 ms. |
| `Flt Env` | Filter envelope depth −4…+4 octaves. |
| `Osc2 Wave` | Oscillator 2 waveform: Sine, Tri, Saw, Square. |
| `Osc2 Semi` | Oscillator 2 coarse tune −24…+24 semitones (+12 = octave up, +7 = fifth, −12 = sub). |

| | |
|---|---|
| `Osc2 Fine` | Oscillator 2 fine tune −50…+50 cents — a few cents against osc 1 beats into a fat unison. |
| `Osc Mix` | Blend 0–100: 0 is oscillator 1 alone (the classic single-osc sound), 100 is oscillator 2 alone. |
| `LFO Rate` | LFO speed 0.1–20 Hz (also the mod-wheel vibrato rate). |
| `LFO Pitch` | LFO → pitch depth 0–2 semitones — fixed vibrato on top of whatever the mod wheel adds. |
| `LFO Cutoff` | LFO → cutoff depth −4…+4 octaves — a filter wobble around Cutoff. |
| `Sub` | Sub oscillator level 0–100, an octave (or two, **Sub Octave**) under oscillator 1; **Sub Wave** Sine or Square. |
| `Noise` | Noise level 0–100; **Noise Color** White or Pink (darker). |
| `Pulse Width` | Width of the Square waves 5–95% (50 = square; narrower is thinner, nasal). |
| `PWM` | LFO → pulse width 0–45% — the classic moving, chorused pulse. |
| `Key Track` | Cutoff follows the keys 0–100%: 0 holds it still, 50 (default) is gentle, 100 tracks an octave per octave. |
| `Vel → Cutoff` | Soft notes up to 4 octaves darker. |
| `Vel → Amp` | How much velocity sets the level, 0–100% (100 = default; 0 plays every note at full level). |
| `LFO Shape` | Sine, Tri, Square or S&H (a new random value each cycle). |
| `LFO Sync` | Free runs at LFO Rate; Sync locks it to the tempo at **LFO Division** (1/4, 1/8, 1/16, dotted, triplets). |
| `LFO Fade` | LFO fade-in 0–5000 ms after each note — vibrato that blooms. |
| `Voice` | Poly (default); Mono — one note at a time, each restrikes, gliding from the last; Legato — overlapping notes slide in without restriking, detached notes start fresh. |
| `Glide` | Glide time 0–2000 ms in Mono and Legato. With a Glide node in front as well, the note glides twice — use one or the other. |
| `Pitch Env` | Pitch at the strike −24…+24 semitones, falling to the note over **Pitch Decay** (1–2000 ms): +12…+24 with ~60 ms is the 808 drop. |
| `Drive` | Saturation into the filter 0–100%, level-matched — grit without getting louder. |

| | |
|---|---|
| `Width` | Stereo spread 0–100%: successive voices sit at alternating places. Mono and Legato stay centred. |
| `Pressure → / Slide →` | Where an MPE note's pressure (channel pressure) and slide (CC 74 on its own channel) go, each with an **Amount** –100…+100%: Off, Level, Cutoff, Reso, Vibrato, Drive, Osc Mix. Default Pressure → Level +100% (up to +6 dB), Slide → Cutoff +50% (±2 octaves). Each note moves only its own voice. |

◉ Every parameter accepts amber mod wires (MIDI In signals, LFO, Attractor, Life stats, etc.).




### Drums

`In: 1× MIDI · Out: — (audio sink)`

tail for claps, a ringing resonator for rims, and a family of percussion picked by note (cowbell 56, tambourine 54, shakers 69/70/82, claves 75, woodblocks 76/77, bongos 60/61, congas 62–64, timbales 65/66, agogô 67/68). Closed hats choke open hats. Renders audio in-plugin — no MIDI output.

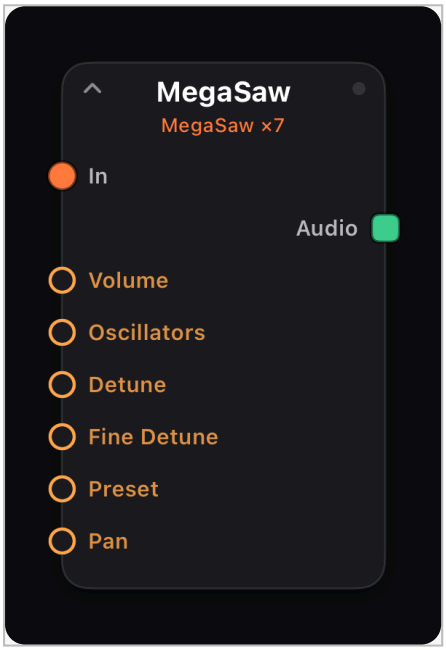
Tune and shape each voice independently, then pick a **Kit** model and set **Tone**, **Snap** and **Drive** for the whole kit. Route a **Drum Seq** branch or **Drum Pads** into **Drums** for an all-in-one groovebox.

## PARAMETERS

| | |
|---|---|
| `Preset` | Kit preset: 33 kits by scene — Studio (Punchy, Boomy, Tight, Acoustic, Big Room), Machines (808, 909, Eighties, Linn, 606, 707), Club (Techno, Garage, House, Minimal, Tech House), Breaks (Glitch, DnB, Jungle, Breakbeat, Big Beat), Hip Hop (Lo-Fi, Trap, Dusty, Boom Bap, Drill, Phonk, Crunk) and Dub (Reggae, Dub, Steppers, Rub-a-Dub, Dancehall). Picking one fills the controls below so you can tweak from there. |
| `Volume` | Master level 0–1. |
| `Per-voice Pit / Dec` | Kick, Snare, Rim, Clap, Cl/Op Hat, Lo/Mid/Hi Tom, Crash, Ride, Perc — each has **Pit** (−12…+12 semitones) and **Dec** (25–400% tail length). |
| `Kit` | The model the voices follow: **Analog** (round and long, pure sines and the metal bank — 808-ish) or **Crunch** (hard pitch drops, brighter noise and saturation through a 12-bit, 26 kHz sampler — the dusty MPC/SP sound). |
| `Tone` | Noise and cymbal brightness 0–100 (50 = as written). |
| `Snap` | The transient click on kicks, snares and toms, 0–100. |
| `Drive` | Per-voice saturation 0–100 on top of each voice's own. |
| `Per-voice Lvl` | Each voice's level 0–100 % — balance the kit. |
| `Per-voice Pan` | Each voice's place in the stereo field (on the panel's **Pan · Tone** page). |
| `Per-voice Tone` | Brightness on top of the kit's Tone, −50…+50, for the snare, clap, hats, crash, ride and |

| | |
|---|---|
| | percussion. |
| `Kick / Snare / Hat / Cymbal / Perc Model` | The Kit model for one group of voices — **Kit** follows the kit's, or **Analog** / **Crunch** on their own (an Analog kit with a Crunch snare). Snare covers snare, rim and clap; Perc the toms and percussion. |
| `Choke` | **Hats** (default): a closed hat closes the open one. **+ Cymbals**: also a crash or ride closes the one still ringing. **Off**: the open hat rings over. |

◉ Every parameter accepts amber mod wires (MIDI In signals, LFO, Attractor, Life stats, etc.).




### MegaSaw

`In: 1× MIDI · Out: — (audio sink)`

Terminal **unison saw stack** — up to **200 detuned sawtooth oscillators per voice** for supersaw leads, trance pads and walls of sound. Renders audio in-plugin; no MIDI output.

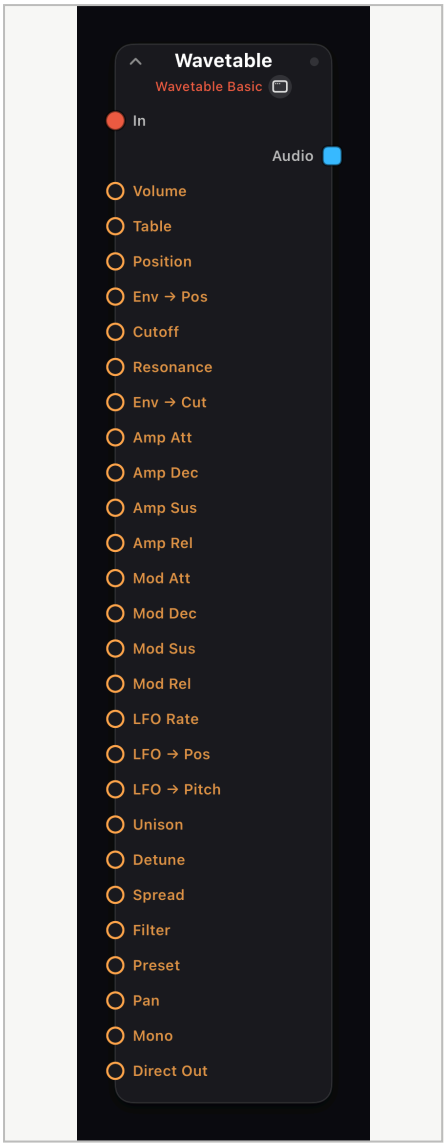
Spread is two-stage: **Detune** sets the coarse pitch spray across the stack, **Fine Detune** adds slow-beating shimmer within it. More oscillators = thicker (and louder — presets lower Volume as the count grows).

Its own **Filter** (SVF, Ladder or Sallen-Key, with an envelope), an **Amp** envelope, stereo **Spread** and a **Sub** make it playable without a Filter node after it. **Soft** — the default — is the stack's original gentle top-end tame.

### PARAMETERS

| | |
|---|---|
| `Preset` | Sound preset: 22 sounds by shelf — Lead (Classic Supersaw, Wide Trance, Soft Lead, Hoover Lead, Thin Lead …), Pads (Massive Wall, Monster (200), Saw Pad, Swell, Dark Saws), Chords (Rave Hoover, Anthem Chord, Stab), Plucks (Trance Pluck, Arp Pluck), Unison Keys, basses (Supersaw Bass, Bass Wall) and FX & Perc (Detuned Chaos, Ocean, Screamer). Picking one fills the controls below so you can tweak from there. |
| `Volume` | Master level 0–1. |
| `Oscillators` | Saws per voice, 1–200. |
| `Detune` | Coarse detune spread 0–100. |
| `Fine Detune` | Slow-beat shimmer within the stack, 0–100. |
| `Filter` | Soft (default: a fixed gentle tame), SVF, Ladder or Sallen-Key. |
| `Cutoff / Resonance` | The filter's cutoff and resonance, 0–100 (not in Soft). |
| `Env → Cut` | Filter envelope depth –4…+4 octaves, over **Flt Att / Dec / Sus / Rel**. |
| `Amp Att / Dec / Sus / Rel` | The amplitude envelope (defaults: the stack's original 5 ms / 120 ms / 65% / 250 ms). |
| `Spread` | Stereo width 0–100%: the stack's saws fan out across left and right. |
| `Sub` | Sine an octave below, 0–100%. |
| `Pressure → / Slide →` | Where an MPE note's pressure (channel pressure) and slide (CC 74 on its own channel) go, each with an **Amount** –100…+100%: Off, Level, Cutoff, Reso, Vibrato, Detune. Default Pressure → Level +100%, Slide → Cutoff +50%. Each note moves only its own voice. |

◉ Every parameter accepts amber mod wires (MIDI In signals, LFO, Attractor, Life stats, etc.).




### Wavetable

`In: 1× MIDI · Out: — (audio sink)`

Terminal **wavetable synth**. Each voice scans a bank of sixteen single-cycle frames — **Position** picks the frame, and the frames between are crossfaded — so one knob sweeps a sine into a pulse, a vowel from A to U, or an FM index from clean to metallic. Every frame is stored band-limited per octave, so high notes stay clean. Stereo out; renders audio in-plugin, no MIDI output.

Eight tables: **Basic** (sine → triangle → saw → square → pulse), **PWM** (a narrowing pulse), **Vowel** (A → E → I → O → U formants), **Sweep** (a resonant peak climbing the harmonics), **FM** (two operators, index 0 → 7), **Distort** (Casio-style phase distortion), **Fold** (a sine through a

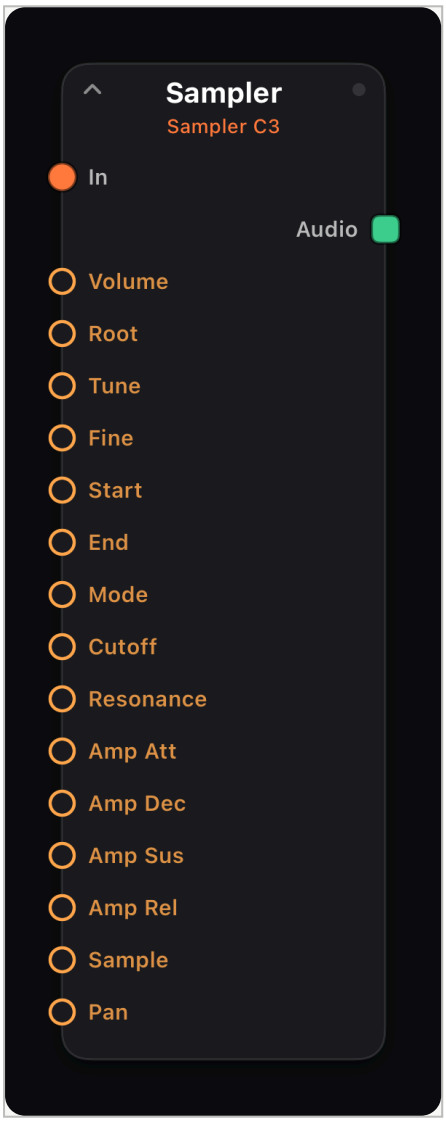
to seven detuned copies across the stereo field. MPE slide nudges the position and pressure leans into the level by default — **Pressure →** and **Slide →** route them elsewhere.

### PARAMETERS

| | |
|---|---|
| `Preset` | Sound preset: 52 sounds by shelf — Bass (PWM Bass, Grit Bass, Vowel Bass, Wobble Table, Donk, Reese Table, Acid Table …), Lead (Sweep Lead, Sync Lead, Formant Lead, Flute, Vox Lead …), Keys (Glass Keys, House Piano, E-Piano, Vibes, Clav Table …), Pads (Vowel Pad, Digital Choir, Evolving Pad, Vox Pad, Spectral Pad …), Plucks (Folded Pluck, FM Bell, Chime, Harp, Mallet, Pluck Table), Chords (Table Stab, Sweep Chord, Vowel Stab, Lo-Fi Chord, Trance Chord) and FX & Perc (Drone, Laser Table, Glitch Hit, Breath, Riser Table, Impact). Picking one fills the controls below so you can tweak from there. |
| `Volume` | Master level 0–1. |
| `Table` | The frame bank: Basic, PWM, Vowel, Sweep, FM, Distort, Fold, Grit. |
| `Position` | Scan position through the table, 0–100%. |
| `Env → Pos` | Mod-envelope depth on the position, –100…+100%. |
| `Cutoff / Resonance` | Resonant lowpass; 100% is fully open. |
| `Filter` | Ladder (default, 24 dB) or SVF (12 dB) — as on the Analog. |
| `Env → Cut` | Mod-envelope depth on the cutoff, –4…+4 octaves. |
| `Amp A/D/S/R` | Amplitude envelope (ms / %). |
| `Mod A/D/S/R` | The mod envelope that drives Env → Pos and Env → Cut (ms / %). |
| `LFO Rate` | 0.1–20 Hz. |
| `LFO → Pos` | LFO depth on the position, 0–100%. |
| `LFO → Pitch` | Vibrato depth, 0–2 semitones (the mod wheel adds more). |
| `Unison` | Oscillators per voice, 1–7. |
| `Detune` | Unison detune spread, 0–100 (±50 cents at 100). |
| `Spread` | Stereo width of the unison stack, 0–100%. |
| `LFO 2` | A second LFO for slow drifts: **Rate** 0.05–20 Hz, **→ Pos** 0–100%, **→ Cut** –4…+4 octaves. |

| | |
|---|---|
| `Sub / Noise` | A sine an octave down and white noise into the filter, 0–100% each. |
| `Vel → Cutoff` | Soft notes up to 4 octaves darker. |
| `Vel → Amp` | How much velocity sets the level, 0–100% (70 = default). |
| `Pressure → / Slide →` | Where an MPE note's pressure (channel pressure) and slide (CC 74 on its own channel) go, each with an **Amount** –100…+100%: Off, Level, Cutoff, Reso, Vibrato, Position. Default Pressure → Level +100%, Slide → Position +50% (±¼ table). Each note moves only its own voice. |

◉ Every parameter accepts amber mod wires (MIDI In signals, LFO, Attractor, Life stats, etc.).



### Sampler

`In: 1× MIDI · Out: — (audio sink)`

Terminal **sample player**: one sample repitched chromatically around its root note, SP-style — no time-stretch, and the character that comes with that. Pick a factory sound with **Sample**, or load your own file from the node inspector; the root note is detected automatically so imports land in tune.

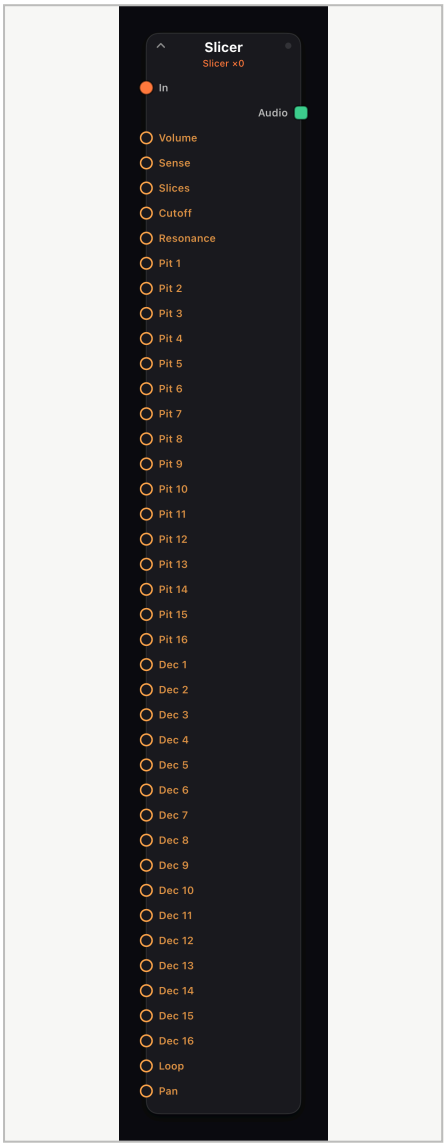
The waveform card on the surface trims the playable region. **Mode** Loop finds a sustain loop inside the region, turning one-shots into playable pads, keys and basses; the Analog's lowpass, amp envelope and filter envelope shape the result (close the cutoff and turn **Flt Env** up for plucks and filter-swept stabs). Slides from a **Glide** node bend the sample like any synth voice.

#### PARAMETERS

| | |
|---|---|
| `Sample` | Factory sound picker (Init keeps the loaded sample). Loading your own file flips it to Custom. |
| `Volume` | Master level 0–1. |
| `Root` | The note the sample plays at original pitch (auto-detected on import). |
| `Tune / Fine` | Coarse ±12 semitones, fine ±50 cents. |
| `Start / End` | Playable region, % of the file. |
| `Mode` | Gate follows note-off · One-Shot plays through · Loop sustains. |
| `Cutoff / Resonance` | Resonant lowpass, as on the Analog. |
| `Amp Att–Rel` | Amplitude ADSR. |
| `Flt Att–Rel` | Filter envelope ADSR, as on the Analog. |
| `Flt Env` | Filter envelope depth –4…+4 octaves around Cutoff (0 = static filter, the default). |
| `Loop` | In Mode Loop: **Auto** (default) finds a loop for you; **Forward** or **Ping-Pong** loop between **Loop Start** and **Loop End** (% of the region), Forward with a **Crossfade** (0–50% of the loop). |
| `Reverse` | Plays Gate and One-Shot notes backwards. |
| `Vel → Cutoff` | Soft notes up to 4 octaves darker. |

| | |
|---|---|
| `Vel → Amp` | How much velocity sets the level, 0–100% (100 = default). |
| `Pressure → / Slide →` | Where an MPE note's pressure (channel pressure) and slide (CC 74 on its own channel) go, each with an **Amount** –100…+100%: Off, Level, Cutoff, Reso, Vibrato. Default Pressure → Level +100%, Slide → Cutoff +50%. Each note moves only its own voice. |

◉ Every parameter accepts amber mod wires (MIDI In signals, LFO, Attractor, Life stats, etc.).




### Slicer

`In: 1× MIDI · Out: — (audio sink)`

and so on. Point one **Drum Seq** lane per slice at its note and the loop becomes a kit you can resequence.

The waveform card shows the markers: tap a slice to hear it, double-tap to add or remove a marker, drag one to nudge. **Sense** re-runs detection — more sensitivity, more slices. Marker positions live in the patch like any other parameter, so edits persist and undo.

After the loops the picker holds **drum kits** — sixteen one-shots, a pad per slice in a fixed order (Kick, Snare, Cl Hat, Op Hat, Clap, Mid Tom, Crash, Ride, Rim, Lo Tom, Hi Tom, Cowbell, Shaker, Clave, Conga, Snare 2), nine rendered from the Drums voices and six sampled from real drum machines. Kit pads answer the GM drum notes the Drums node answers (36 kick, 38 snare, 42 closed hat…), not chromatic slice notes, so a **Drum Grid** on its default notes plays a kit exactly as it plays the Drums node.

### PARAMETERS

| | |
|---|---|
| `Loop` | Factory loop and kit picker (Init keeps the loaded sample). Imported loops are sliced on load. |
| `Volume` | Master level 0–1. |
| `Sense` | Onset-detection sensitivity 0–100 — re-slices the loop. |
| `Slices` | Active slice count (written by detection, editable). |
| `Cutoff / Resonance` | Resonant lowpass over all slices, as on the Sampler. |
| `Per-slice Pit / Dec` | Each slice has **Pit** (−12…+12 semitones) and **Dec** (25–400% — under 100 chokes the hit early). |
| `Per-slice Lvl / Pan` | Each slice's level 0–100% and place in the stereo field. |
| `Per-slice Rev` | Plays the slice backwards. |
| `Per-slice Choke` | Choke group Off or A–D: a slice cuts the others in its group (an open hat closed by the closed one). |
| `Flt Env, Flt Att–Rel` | A filter envelope over Cutoff, −4…+4 octaves (0 = static, the default). |

◉ Every parameter accepts amber mod wires (MIDI In signals, LFO, Attractor, Life stats, etc.).




### AU Instrument

`In: 1× MIDI · Out: — (audio sink)`

A **hosted third-party AUv3 instrument** — any instrument plugin installed on the device plays the MIDI routed into this node, mixed into Konstrukt's output like a built-in instrument. Pick the plugin in the inspector (**Choose…**), then open its own window from there to edit its sound; the plugin's settings are saved with the patch.

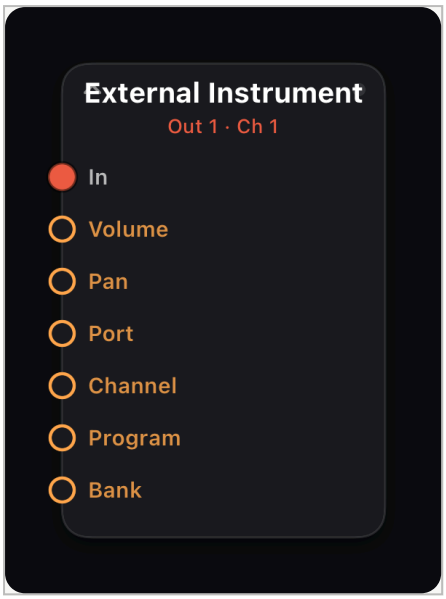
Its parameters appear as node parameters you can bind and modulate; a frequency or time parameter with a wide range (a 20 Hz–20 kHz cutoff) gets a **logarithmic taper** automatically — a bound knob spends equal travel on equal ratios, so the sweep feels like the plugin's own knob.

**Standalone app only.** A plugin can't host other plugins, so inside a host like AUM or Logic the node stays in the patch but is silent — route that MIDI to a track instead.

#### PARAMETERS

| | |
|---|---|
| `Volume` | Output level of the hosted plugin, 0–1. |

◉ Every parameter accepts amber mod wires (MIDI In signals, LFO, Attractor, Life stats, etc.).




### External Instrument

`In: 1× MIDI · Out: — (leaves on a MIDI output port)`

A **synth, drum machine or another app on a MIDI output**, as a channel. Like an instrument it is a terminal — wire lanes and keyboards into it and nothing flows back into the graph — but instead of sounding, everything leaves on its **Port** forced onto its **Channel** (**MPE** keeps each note's own channel). On the mixer it gets a channel beside the audio channels: a fader and pan that send **CC 7 / CC 10**, mute and solo that hold the gear silent (held notes are released), and **Program** / **Bank** pickers that send a program change.

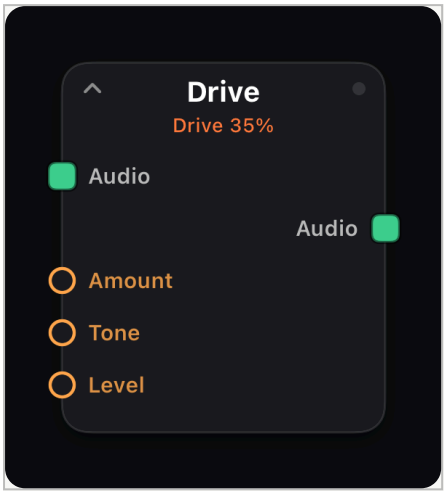
One node per piece of gear. Prefer it over MIDI Out's channel connectors whenever the thing on the other end has a name: the mixer, scenes, surface bindings and MIDI Learn all treat it as a mixer channel.

#### PARAMETERS

| | |
|---|---|
| `Volume` | Sent as CC 7 (0–1 → 0–127) whenever it moves. |
| `Pan` | Sent as CC 10 (–100…100 → 0–127) whenever it moves. |
| `Port` | The MIDI output port the gear is on (Out 1–4). |
| `Channel` | MPE (per-note channels pass through) or Ch 1–16 (forced). |
| `Program` | — (nothing sent) or 1–128; sent as a program change when it moves. |
| `Bank` | — or 1–128; sent as bank select MSB (CC 0) before the program. |

◉ Every parameter accepts amber mod wires (MIDI In signals, LFO, Attractor, Life stats, etc.).

# Audio FX — drive, filter, delay, reverb — stereo audio chains




### Drive

`In: 1× Audio · Out: 1× Audio`

**Saturation for your instruments.** Audio FX nodes carry **audio**, not MIDI: wire an instrument's Audio output into the effect's input, chain effects into each other, and the last effect in the chain feeds the speakers. Instruments left unwired play dry, straight to the output — so effects apply only to what you route through them, in **stereo**.

Drive pushes its input into a shaper in one of five characters: **Soft** is a smooth tanh — gentle warmth at low Amount, full-on fuzz at high; **Tube** rounds the two half-waves differently for even harmonics; **Tape** saturates the highs first and stays soft; **Fuzz** is a hard clip; **Fold** is a wavefolder — whatever passes the top comes back down, metallic and synthy. **Tone** is a post-shaper lowpass that tames the added brightness; **Mix** blends the dry signal back in for parallel drive on drums.

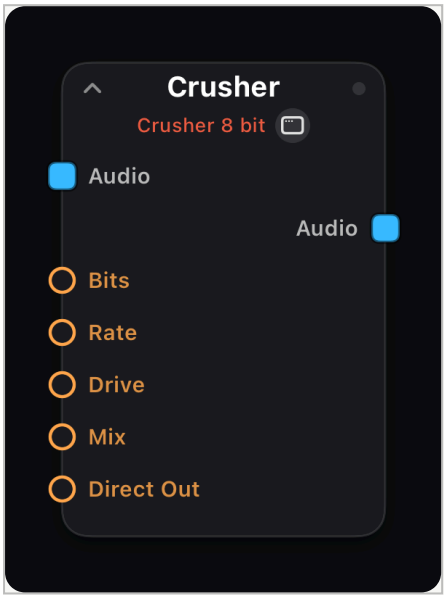
The output is matched to the input's level: a driven channel never rises above its own peaks, it gets denser — turn Level up if you want it louder too. **Oversample** runs Tube, Tape, Fuzz and Fold at twice the sample rate so the grit stays harmonic instead of folding back as aliasing — worth it on bright sources, off by default to keep the CPU light.

#### PARAMETERS

| | |
|---|---|
| `Amount` | Input gain into the shaper, 0–100 %. |
| `Tone` | Post lowpass, dark → bright (500 Hz–12 kHz). |
| `Level` | Output trim 0–100 %. |

| | |
|---|---|
| `Mode` | **Soft** · **Tube** · **Tape** · **Fuzz** · **Fold**. |
| `Mix` | Dry/wet, 0–100 %. |
| `Oversample` | Off / 2× — Tube, Tape, Fuzz and Fold at twice the rate (Soft never is). |

◉ Every parameter accepts amber mod wires (MIDI In signals, LFO, Attractor, Life stats, etc.).




### Crusher

`In: 1× Audio · Out: 1× Audio`

**Bit crusher and sample-rate reducer** — the grit of an old sampler. **Rate** holds each sample for a while with no anti-aliasing, so high notes fold back into metallic overtones; **Bits** rounds what's left to fewer steps, and at low settings quiet tails drop out entirely, which chops reverbs and decays into gated stutters.

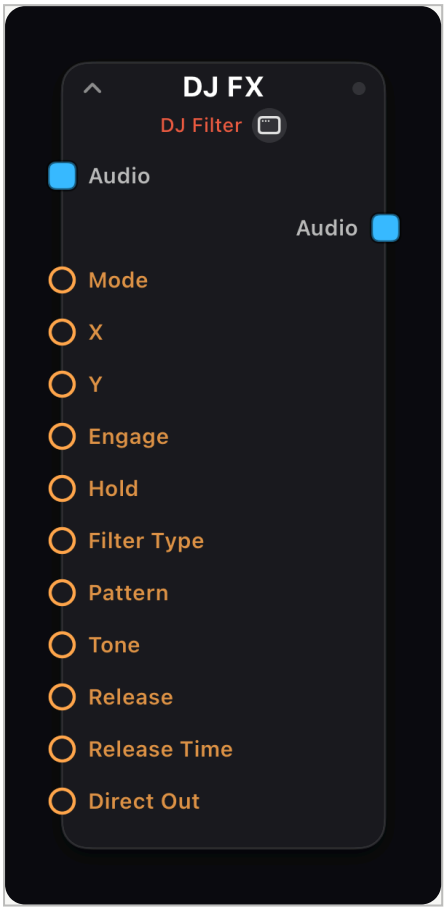
Bits sweep smoothly between whole values, so a knob or an LFO on Bits is a usable effect, not a stepped switch. Put it on a break for lo-fi hip hop, on hats and percussion for IDM, or on a send and blend it in.

#### PARAMETERS

| | |
|---|---|
| `Bits` | Resolution 1–16 bits (fractional values sweep). |
| `Rate` | Sample-and-hold rate, 200 Hz – 44.1 kHz. |
| `Drive` | Input gain into a hard clip, 0–100 %. |

| | |
|---|---|
| `Mix` | Dry/wet, 0–100 %. |

◉ Every parameter accepts amber mod wires (MIDI In signals, LFO, Attractor, Life stats, etc.).




### DJ FX

`In: 1× Audio · Out: 1× Audio`

**A performance effect played from an XY pad**, Kaoss style. Its panel is the pad: **touch** it to engage the effect, move the finger for **X** and **Y**, **lift** to let go — or turn on **Hold** and the pad stays engaged until Hold goes off. Not engaged, the sound passes untouched. One **Mode** at a time:

**Filter** — X sweeps a lowpass to the left of the centre and a highpass to the right, off in the middle; Y is resonance. **Repeat** — the beat-grid slice under your finger (X: 1/4 … 1/32) plays through, then repeats; Y lowers each repeat's pitch. **Tape Stop** — the sound slows to a halt over X (1/8 beat … 4 beats) and spins back up on release; Y is wet. **Gater** — chops in time at X's rate in a pattern; Y is depth. **Echo Out** — a dub throw: what you play while the pad is held goes into an echo (X: 1/16 … 1/2, Y: feedback) that rings on after you let go.

**Riser** — a build-up of its own: noise through a rising resonant band-pass, a high-pass climbing on the input and, with Tone up, a sine rising two octaves. X picks 1, 2, 4 or 8 bars, and the riser

build (0) against the noise (100). It plays even with nothing coming in, and runs at the session tempo while the transport is stopped.

**Release** (beside Hold) decides what letting go does, like a Kaoss Pad's FX Release. On, the effect tails off over the **Release Time**, 1/8 beat … 4 beats (1 beat to start): the Filter glides back to off, Repeat keeps repeating as it fades, Tape Stop spins back up, the Gater's depth fades and the Riser fades out; Echo Out's echoes ring on by themselves, whatever the time. Off, the pad lets go dry at once and cuts any tail. Touching the pad during a release picks the effect back up. Release is for the lift: switching modes while held changes over at once (Echo Out's echoes still ring on).

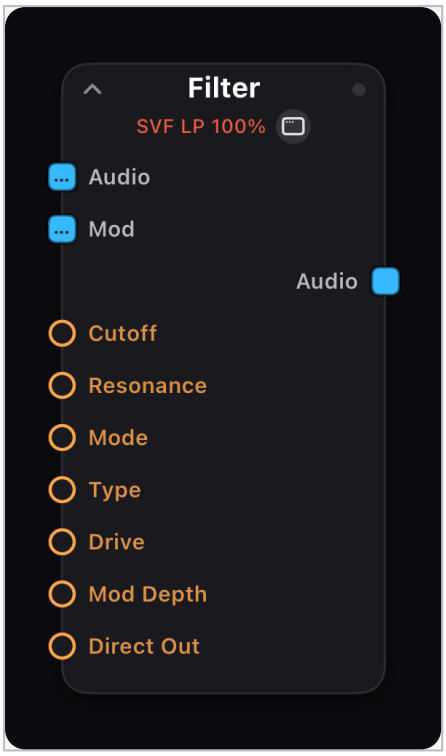
**Two pads** side by side (the **Pads** switch: one or two): pad **B** is a second DJ FX with its own Mode, X / Y, Hold and Release, after pad A — a filter on one hand and a stutter on the other. It starts on Repeat beside A's Filter.

X, Y and Engage are ordinary parameters: a surface knob, XY pad or button, MIDI learn or a mod wire can play them. Put it last on a channel, or on the master for the whole mix.

### PARAMETERS

| | |
|---|---|
| `Mode` | Filter, Repeat, Tape Stop, Gater, Echo Out, Riser. |
| `X / Y` | The pad, 0–100; what they do follows the Mode (above). |
| `Engage` | On while the pad is held; Off passes the input untouched. |
| `Hold` | Latches the pad: a touch engages it until Hold goes off. |
| `Filter Type` | Filter mode: SVF, Ladder (default) or Sallen-Key. |
| `Pattern` | Gater mode: Straight, Offbeat or Trance. |
| `Tone` | Riser mode: the rising sine's level, 0–100 % (0 = off). |
| `Release` | On (default): letting go tails the effect off over the Release Time. Off: it lets go dry. |
| `Release Time` | 1/8, 1/4, 1/2, 1 (default), 2 or 4 beats. |
| `Pads` | 1 or 2 (default). Back to 1 lets go of pad B. |
| `B Mode … B Release Time` | Pad B's own set of the above, after pad A; B Mode starts on Repeat. |

◉ Every parameter accepts amber mod wires (MIDI In signals, LFO, Attractor, Life stats, etc.).




### Filter

`In: 1× Audio + 1× Mod (audio-rate CV) · Out: 1× Audio`

A stereo **virtual-analog filter** for whatever you wire into it, with three classic circuits modelled zero-delay-feedback style, so the cutoff tracks exactly and sweeps stay smooth. Bind a knob — or an LFO, via a modulation wire onto the Cutoff row — for classic sweeps.

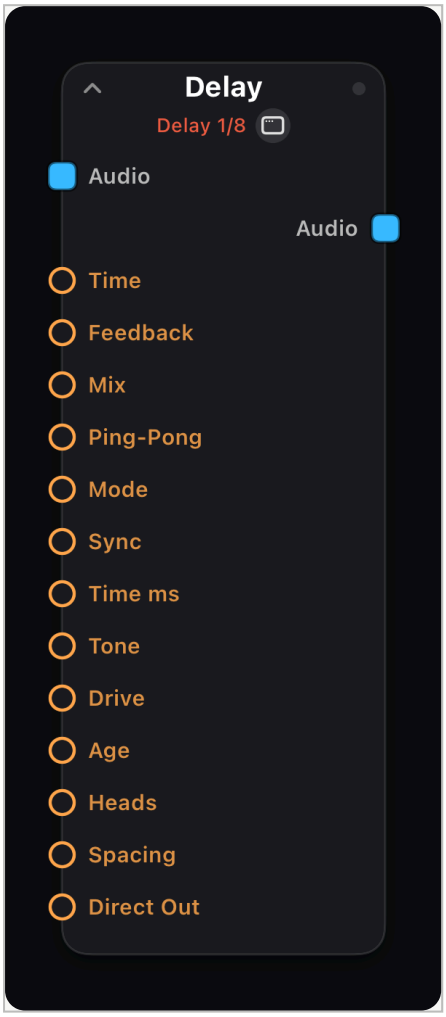
**SVF** is the clean, precise 12 dB state-variable. **Ladder** is a 24 dB transistor ladder: steeper, warmer, its bass thins as Resonance rises, and it self-oscillates near full Resonance. **Sallen-Key** is a 12 dB MS-20-style core whose feedback clips — resonance screams, then compresses, and it self-oscillates too.

#### PARAMETERS

| | |
|---|---|
| `Type` | SVF (12 dB) / Ladder (24 dB) / Sallen-Key (12 dB). |
| `Mode` | Lowpass / Bandpass / Highpass, on every type. |
| `Cutoff` | 0–100 %, exponential sweep 20 Hz – 20 kHz. |
| `Resonance` | Peak at the cutoff, 0–100 %; Ladder and Sallen-Key sing on their own above ~95 %. |
| `Drive` | Input saturation, 0 % clean → +24 dB into a soft clipper, level-compensated. |

| | |
|---|---|
| `Mod Depth` | How far a signal on the **Mod** input moves the cutoff: 0–8 octaves at full scale. Wire any audio into Mod for an audio-rate sweep. |

◉ Every parameter accepts amber mod wires (MIDI In signals, LFO, Attractor, Life stats, etc.).




### Delay

`In: 1× Audio · Out: 1× Audio`

**A stereo delay in three voices. Digital** is the clean line; **Tape** is a tape echo — three playback heads, wow and flutter, saturation and hiss, and a Feedback that runs past 100 % into a held, self-oscillating dub echo; **Analog** is a bucket-brigade delay, every repeat darker and grainier than the last.

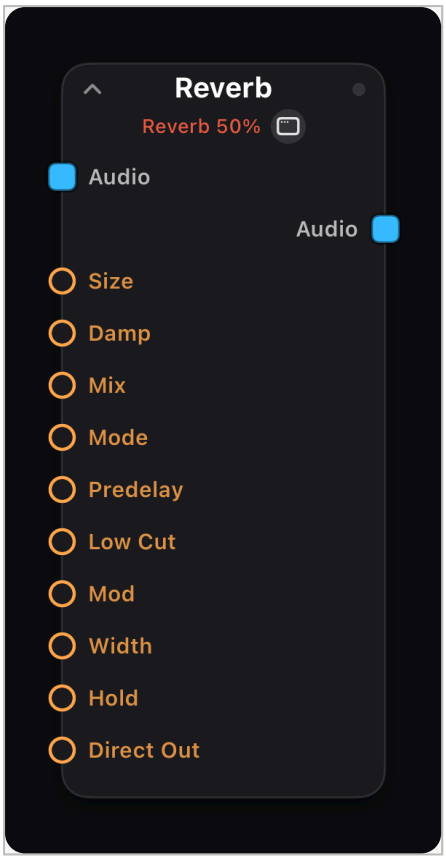
Time is a clock division that follows the transport — or, with **Sync** off, milliseconds for slap-back and dub throws; changing it glides the delay line tape-style instead of clicking. **Ping-Pong**

crosses the feedback so repeats bounce left ↔ right. Switching the mode ducks the echoes for a moment rather than clicking.

### PARAMETERS

| | |
|---|---|
| `Time` | Delay interval as a clock division (`4/1` … `1/64`, dotted and triplet variants included). |
| `Feedback` | Repeat level, 0–95 % (Tape: up to 120 %). |
| `Mix` | Dry/wet, 0–100 %. |
| `Ping-Pong` | Off / On — cross-feed the repeats between channels. |
| `Mode` | **Digital** · **Tape** · **Analog**. |
| `Sync` | **Sync** — Time is a division; **Free** — **Time ms** instead (1–2000 ms). |
| `Tone` | 50 % is neutral. Digital: lower darkens the repeats, higher thins them; Tape and Analog: the loop's brightness. |
| `Drive` | Tape and Analog: how hard the repeats saturate. |
| `Age` | Tape: wow, flutter and hiss; Analog: clock drift and noise. The noise only rides the echoes — silence stays silent. |
| `Heads` | Tape: which playback heads sound — 1, 2, 3 or a combination. |
| `Spacing` | Tape: the heads' distances as multiples of Time — 1·2·3, 1·1.5·3, 1·1.25·1.75, 1·2·4. |

◉ Every parameter accepts amber mod wires (MIDI In signals, LFO, Attractor, Life stats, etc.).




### Reverb

`In: 1× Audio · Out: 1× Audio`

**A stereo reverb in four rooms. Plate** — a diffused, gently modulated figure-eight tank with a wide, smooth tail; **Hall** — the same tank stretched, with early reflections and longer decays; **Gated** — the 80s gated snare: a dense tail that cuts **Hold** ms after the input falls quiet; **Vintage** — an early digital box, 12-bit at half rate, grainy and dark.

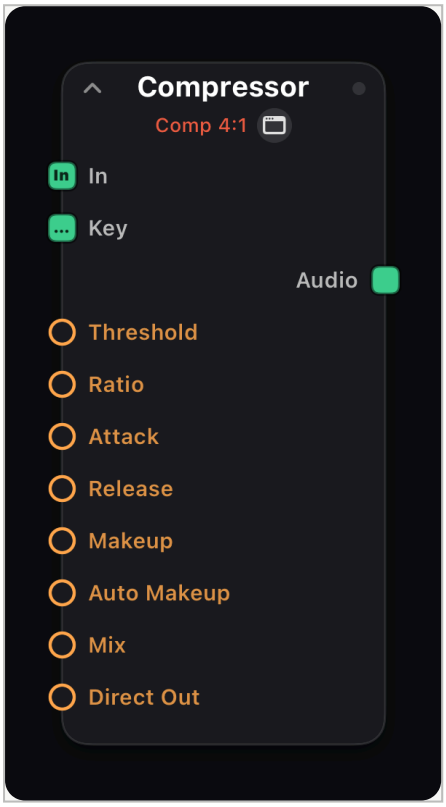
Short rooms to long washes via **Size**; **Damp** rolls highs off the tail. Chain it after a Delay for classic dub tails.

#### PARAMETERS

| | |
|---|---|
| `Size` | Tail length / room size, 0–100 %. |
| `Damp` | High-frequency damping of the tail, 0–100 %. |
| `Mix` | Dry/wet, 0–100 %. |
| `Mode` | **Plate** · **Hall** · **Gated** · **Vintage**. |
| `Predelay` | 0–250 ms before the room answers. |

| | |
|---|---|
| `Low Cut` | 20–1000 Hz high pass into the room (100 Hz by default) — keep the bass out of the tail. |
| `Mod` | Movement in the tail, 0–100 % (50 by default). |
| `Width` | 0 % mono … 100 % full stereo. |
| `Hold` | Gated: how long the gate stays open after the input falls quiet, 20–1000 ms. |

◉ Every parameter accepts amber mod wires (MIDI In signals, LFO, Attractor, Life stats, etc.).




### Compressor

`In: 1× Audio + Key (sidechain) · Out: 1× Audio`

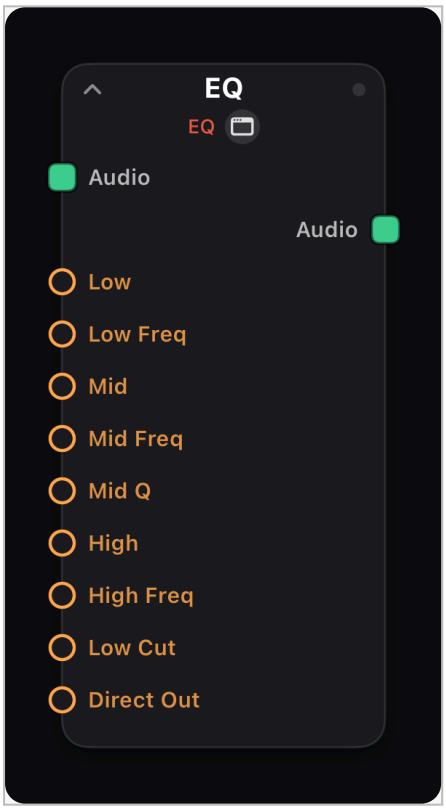
**Evens out levels — or pumps them.** A stereo-linked, feed-forward compressor with a soft knee: everything over **Threshold** is turned down by **Ratio**, with **Attack** and **Release** setting how fast it grabs and lets go. Its panel draws the curve and a live **gain-reduction meter**.

The **Key** input is a sidechain: wire another instrument's Audio output into it (or pick it under *Sidechain from* in the insert's menu on the mixer) and the key's level, not the input's, drives the gain. A kick keyed into a pad's compressor is the classic pumping pad. The key is only heard by the detector — with nothing on In, the compressor is silent. **Mix** below 100 % blends the dry signal back in: parallel compression.

### PARAMETERS

| | |
|---|---|
| `Threshold` | Level where compression starts, –60…0 dB. |
| `Ratio` | 1:1 (off) … 20:1 (limiting). |
| `Attack` | How fast it reacts, 0.1–100 ms. |
| `Release` | How fast it recovers, 10–1000 ms. |
| `Makeup` | Output gain after compression, 0–24 dB. |
| `Auto Makeup` | Off / On — adds back half the reduction a full-scale peak would get. |
| `Mix` | Dry/compressed blend, 0–100 %. |

◉ Every parameter accepts amber mod wires (MIDI In signals, LFO, Attractor, Life stats, etc.).




**EQ**

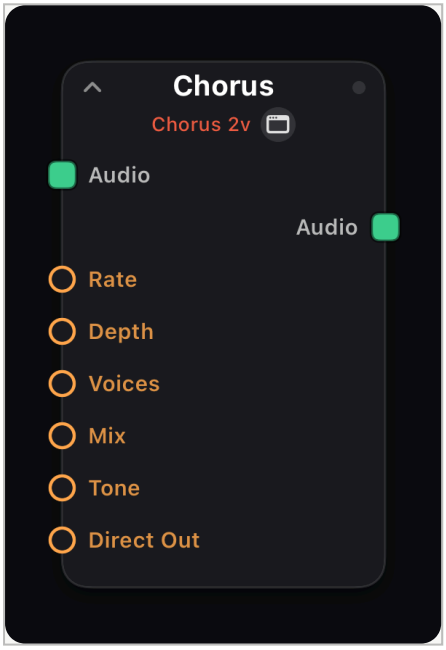
`In: 1× Audio · Out: 1× Audio`

**A 3-band channel EQ.** A low shelf, a peaking mid with adjustable width (**Q**) and a high shelf, each ±15 dB, plus a **Low Cut** (a 12 dB/oct highpass) to clear rumble and mud. The panel draws the combined response curve. Sweeps glide, so an LFO on a frequency or gain doesn't click.

PARAMETERS

| | |
|---|---|
| `Low / Low Freq` | Low shelf gain ±15 dB, corner 30–500 Hz. |
| `Mid / Mid Freq / Mid Q` | Peak gain ±15 dB, centre 200 Hz–8 kHz, width 0.3–8. |
| `High / High Freq` | High shelf gain ±15 dB, corner 1.5–16 kHz. |
| `Low Cut` | Highpass 20–500 Hz; all the way down is Off. |

◉ Every parameter accepts amber mod wires (MIDI In signals, LFO, Attractor, Life stats, etc.).




### Chorus

`In: 1× Audio · Out: 1× Audio`

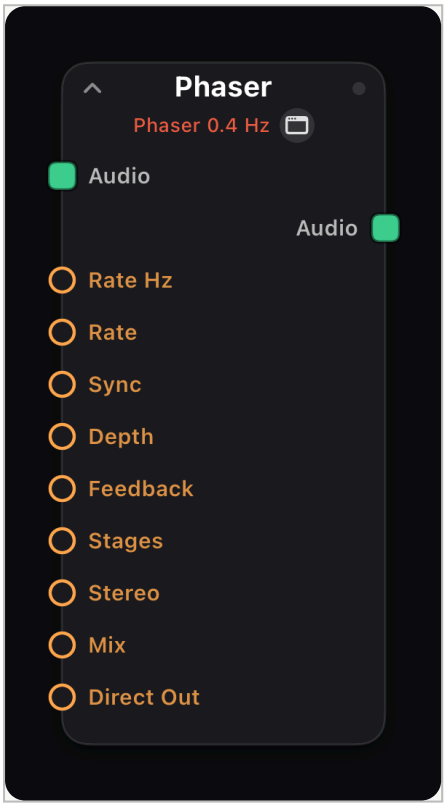
**Width and movement for pads and keys.** One to three slowly modulated copies of the signal (7–25 ms), with the left and right channels modulated a quarter-cycle apart — so a mono synth comes out wide. **Tone** darkens the chorused signal, Juno-style.

PARAMETERS

| | |
|---|---|
| `Rate` | Modulation speed, 0.05–5 Hz (free-running). |

| | |
|---|---|
| `Depth` | Modulation amount, 0–100 %. |
| `Voices` | 1 / 2 / 3 modulated copies. |
| `Mix` | Dry/wet, 0–100 %. |
| `Tone` | Wet-signal brightness, 0–100 %. |

◉ Every parameter accepts amber mod wires (MIDI In signals, LFO, Attractor, Life stats, etc.).




### Phaser

`In: 1× Audio · Out: 1× Audio`

**The swirling sweep.** 4 to 12 allpass stages swept between about 100 Hz and 4 kHz carve moving notches into the sound; **Feedback** sharpens them. In **Sync** the sweep follows the transport at a clock division, landing the same way every loop — a synced phaser on a drum loop or a Slicer break is the classic use. **Stereo** offsets the right channel's sweep.

#### PARAMETERS

| | |
|---|---|
| `Rate Hz` | Sweep speed in Free mode, 0.02–10 Hz. |

| | |
|---|---|
| `Rate` | Sweep length in Sync mode, a clock division. |
| `Sync` | Sync (transport) / Free (Hz). |
| `Depth` | Sweep width, 0–100 %. |
| `Feedback` | Notch sharpness, 0–90 %. |
| `Stages` | 4 / 6 / 8 / 12 allpass stages — more notches. |
| `Stereo` | Right-channel sweep offset, 0–100 %. |
| `Mix` | Dry/wet, 0–100 %; 50 % gives the deepest notches. |

◉ Every parameter accepts amber mod wires (MIDI In signals, LFO, Attractor, Life stats, etc.).



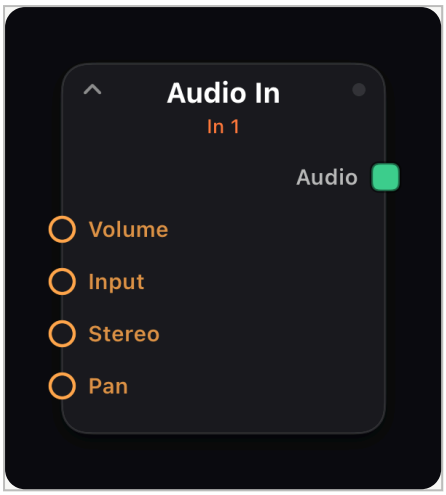

### AU Effect

`In: 1× Audio · Out: 1× Audio`

A **hosted third-party AUv3 effect** in the audio chain — any effect plugin installed on the device processes the audio wired into it, exactly like Drive or Reverb. Pick the plugin in the inspector (**Choose…**) and open its own window from there; its settings are saved with the patch. Music effects (plugins that also take MIDI) receive the MIDI routed through the node.

**Sidechain.** A plugin with a second input bus grows a **Key** port on the node: wire any channel into it (or pick **Sidechain from** on the insert chip or bus strip in the mixer) and the plugin ducks, gates or follows that signal. A Key wire carries the whole signal, counts as neither a send nor an insert, and doesn't take the source off the master.

**Standalone app only.** Inside a host like AUM or Logic the node stays in the patch but passes audio through untouched.




### Audio In

`In: — · Out: 1× Audio`

**Live audio input** — the microphone or a channel of your audio interface as an audio *source*. It has no MIDI ports: wire its Audio output into effects or straight to Audio Out, and it gets a mixer channel like any instrument (meter, fader, pan, mute/solo, inserts, sends). Run a guitar through Drive and Delay, a synth through Reverb, or a mic through a hosted AU effect.

**Input** picks the hardware channel (1 = the first input of the interface, or the built-in mic); **Stereo** takes the pair starting there. The app asks for microphone access the first time a patch carries an Audio In node, and only switches the audio session to record mode while one exists.

**Standalone app only.** Inside a host like AUM or Logic the node stays in the patch but is silent — route the host's input to the track instead.

#### PARAMETERS

| | |
|---|---|
| `Volume` | Input gain 0–1 (default 1). |
| `Input` | Hardware input channel 1–16. |
| `Stereo` | Off: the one channel, centered. On: the channel pair starting at **Input**. |
| `Pan` | Mixer pan −100…100 (balance law: centre = unity). |

◉ Every parameter accepts amber mod wires (MIDI In signals, LFO, Attractor, Life stats, etc.).




### Audio Out

`In: 1× Audio · Out: —`

**The audio terminal**, mirroring MIDI Out: every instrument and effect chain wires its Audio output here to be heard. The app adds it (and wires up loose chain ends) automatically whenever a patch has audio nodes, so nothing goes silent by accident.

Wire an instrument into **both** an effect and Audio Out and it plays wet and dry in parallel — a classic dry/wet send.

On a multi-out interface, **Output** picks the stereo pair the master leaves on; channels and buses can leave on their own pair with **Direct Out** (mixer strip menu › Output). A pair the current route lacks folds back to the master.

PARAMETERS

`Output` Which stereo pair of a multi-out interface the master leaves on (1–2 by default).

◉ Every parameter accepts amber mod wires (MIDI In signals, LFO, Attractor, Life stats, etc.).

## Utility — MIDI monitor



**Monitor**

`In: 1× MIDI · Out: 1× MIDI (unchanged)`

Zero-latency **debug tap** — MIDI passes through unchanged while messages log to the surface Monitor readout or a bound **Note Scope**. Splice anywhere the stream goes dark.

### Control catalog

Each entry shows the control as it appears on the performance surface. Add controls from the header **+** menu in edit mode, then drag into place and open the inspector (sliders icon) to bind.

- **Caption** — `Node · Parameter` when bound to the graph; `CC n` when sending raw MIDI to your synth.
- **Performance controls** (keyboards, pads) inject notes into a `keyboardSource` graph node — route that branch independently.
- **Visualizers** attach to a specific node and update live from the patch.

## Adjust & trigger — knobs, faders, XY, buttons

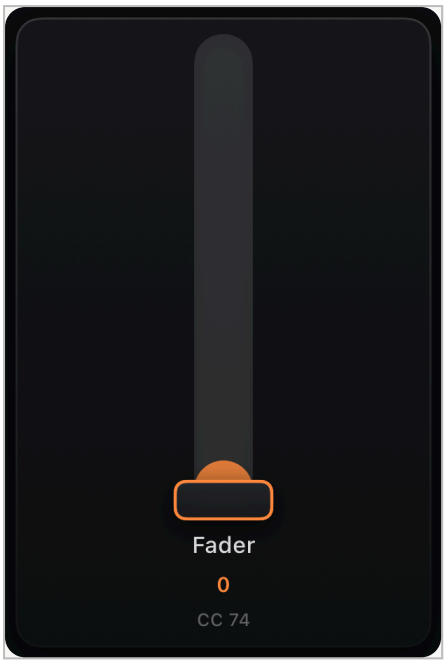



**Knob**

`Binding: node parameter or CC · drag vertically`

Rotary control for any bound parameter or raw CC. Values display musically when the target is a note, clock division or scale name.

Resize in edit mode; color via **Appearance** in the inspector.




### Fader

`Binding: node parameter or CC · drag vertically`

Vertical slider — same binding model as a knob, better for long throws (filter sweeps, levels).

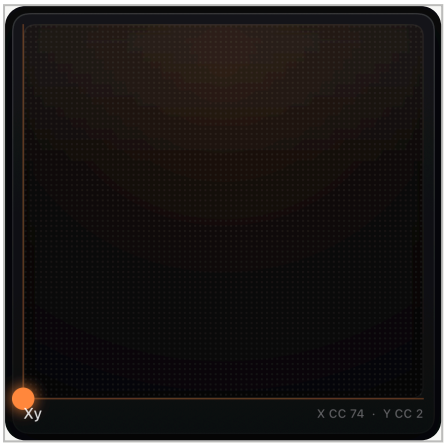



### Selector

`Binding: indexed node parameter (or CC) · tap / scroll to pick`

Touch-friendly picker for **indexed / named** parameters — waveform, scale, mode, chord type, clock division. It adapts to the option count: a **segmented pill bar** for short lists, a **scroll-wheel drum** for long ones.

Prefer it over a knob whenever the parameter is a list of choices rather than a sweepable range. Short enums also appear as inline pills in the node inspector.



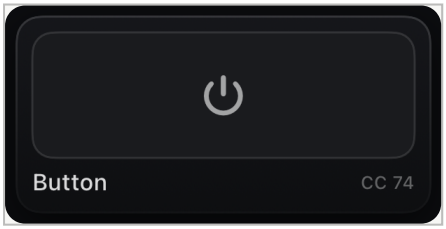

### XY Pad

`Binding: X and Y each → parameter or CC · drag the puck`

Two-axis control. Set **X Axis** and **Y Axis** independently in the inspector — mix filter cutoff and resonance, or drive two node parameters at once.

**Send Note** (inspector) plays a note while the pad is touched. **Record & Loop** adds record / play chips: tap ●, draw one gesture, and it loops — synced to whole beats or free-running.

Every new pad has its own **XY Pad** graph node: wire its MIDI output into a synth to play a note while touched, and **X**, **Y** and **Gate** to any parameter.

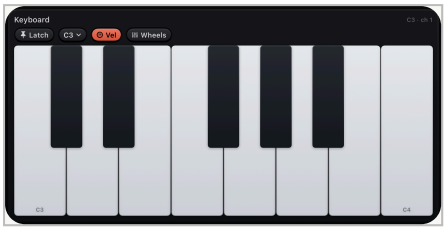



### Button

`Binding: parameter or CC · sends max on, min off`

**Toggle** latches on/off; **Momentary** is active only while held. Use for gates, mode switches, or momentary CC bursts.

# Performance — keyboards and pads




### Keyboard

`Emits: notes · own graph branch via keyboardSource`

Piano keyboard — slide across keys for glissando. **Octave** menu (C−1…C7) sets the range, **Octaves** how many each row shows (Auto fits the width; a tall card stacks rows); **Latch** on the toolbar sustains notes. **Velocity Sensitive** (inspector) maps vertical touch to dynamics.

Each keyboard is a graph node. Wire its output through processors or to a specific **MIDI Out** channel.



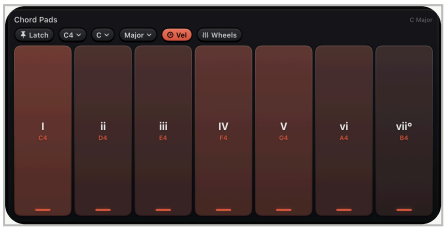

### Drum Pads

`Emits: GM drum notes · channel 10 by default`

4×4 pad grid mapped to General MIDI percussion (kick, snare, hats…). Set **Channel** in the inspector for non-GM rigs.

Switch **Layout** to **Chromatic** in the inspector for a pad grid rising semitone-by-semitone from **Base Note** at the bottom-left pad — for samplers, slicers, and gear with its own drum maps.

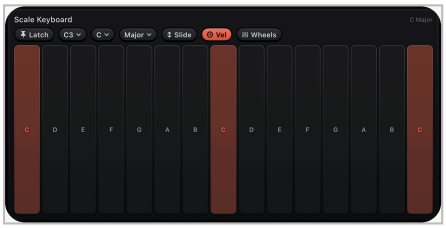
**Custom** layout maps every pad yourself: tap a pad in the inspector's mini grid to set its note and label, or **Learn** it from a hardware pad hit. Pads start from what the previous layout showed, duplicates are allowed, and **Reset** returns to the GM map.




### Chord Pads

`Emits: stacked chord per scale degree`

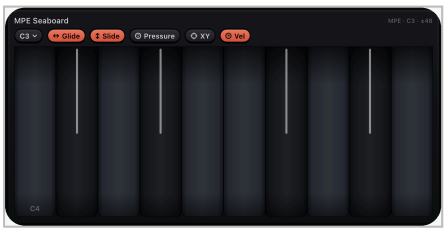
One pad per diatonic degree — plays a triad or 7th from **Root** + **Scale**. **Voices** picks triad vs seventh. Toolbar: latch, octave, root and scale.




### Scale Keyboard

`Emits: in-scale notes only`

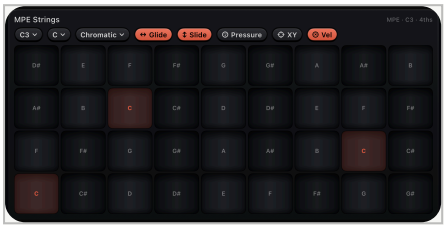
Every visible key is in **Root** + **Scale** — no wrong notes. Good for live performance and teaching patches. **Octaves** per row on the toolbar (two by default, Auto fits the width). Toolbar **Slide**: a held finger's height on its key sends CC 74. **MPE** gives each finger its own channel: glide bends through the scale, slide and pressure are per note.



### MPE Seaboard

`Emits: per-note MPE · glide, slide, pressure`

Continuous ribbon surface: horizontal **glide** (pitch bend per finger), vertical **slide** (CC 74), and **pressure**. Routes to MPE-capable synths on per-note channels.




### MPE Strings

`Emits: per-note MPE · grid tuned in fourths`

Linnstrument-style grid: rows in fourths, columns chromatic. Same MPE expression as Seaboard with a different layout.

## Sequencer — step grids and piano roll




### Sequencer Lane+

`Binding: sequencer lane node · hit, note, deg or val prefix`

The default sequencer-lane control: a 16-step row with a small **layer picker** above it — **Pattern** (hits, pitches, chords or CC values, same tap/drag/microtiming gestures as a plain Step Grid), **Vel** (per-step velocity as a % of the lane's Velocity — drag to shape accents and ghost notes; hidden on CC lanes), **Prob** (per-step probability, drag to set), **Tie** (tap to tie a step to the previous note's gate — hidden on lanes without ties, like CC Seq), and **Cond** (tap to cycle the step's trigger condition) and, on Drum / Step lanes, **Ratchet** (×1…×4 retriggers inside the step — tap to cycle, drag to set). A moving playhead tracks every layer, and every modifier layer keeps the sequence readable: steps the pattern fires are outlined in the lane's colour while

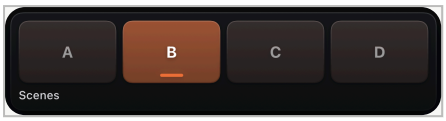
silent steps dim, so a probability or condition lands on the right hit without switching back to Pattern.

On a **Chord Seq** lane the pattern layer is the **Progression** and its cells are **named chord pads** in shades of the lane colour by harmonic function (home I/iii/vi, lighter ii/IV, deeper V/vii). Each pad shows its chord symbol ("C", "Am", "G7") under its roman numeral, following the node's Root / Scale / Chord knobs live. A tied step draws as the held chord's tail. Pick a chord from the **pen palette** above the steps (I…VII in the lane's key, plus a **7th** switch; short lanes carry it as a chip menu), then tap a step to place it — tapping the pen's own chord rests the step — or sweep sideways to paint it along. Drag a pad up or down to step its chord through the degrees; the pen follows. The auto-attached lane fits itself to the active steps, so a 4-chord progression gets four big pads. Under each pad a **Tie** button holds the chord before through the step and a voicing button cycles Root → 1st → 2nd → 3rd (lanes too narrow for the buttons keep Tie and Voicing as layers). Chord lanes swap Cond and Ratchet for a **Rhythm** layer: a second grid, with its own rate and length chips, that fires the chord the current step selected — empty, every chord step plays itself; set hits (tap, sweep to paint, drag up/down for the accent) and the progression walks at Rate while the rhythm stabs it, on 2 and 4 or every offbeat eighth. A **Strum** layer sets each step's roll (taller = longer, 0–100 % of the step; tap steps through none → 10 → 25 → 50 %), with direction and curve chips beside it.

The **A·B·C·D chips** beside the picker are the node's four pattern banks: tap one to edit that bank and switch the sequencer to it at the next pattern wrap. By default the lane **auto-follows** whichever bank is sounding (watch a Chain cycle); tapping a bank chip pins it for manual editing and the **eye chip** re-engages follow. A dot marks the sounding bank when it differs from the one shown. The **◀ ▶ chips** rotate the shown bank one step left or right, every layer travelling with its step (one undo). An Analog Seq row adds a **range** chip (1 / 2 / 3 octaves) that sets what its knobs span.

Drum and Step Seq lanes also **record**: the **● chip** arms **Step Record** (each note lands on the red cursor step and advances it; **Rest** writes a rest; tap a cell to move the cursor) or **Live Record** (notes snap to the nearest step of the sounding bank as the transport runs, velocity and timing feel kept, held notes tied; **Replace** clears the bank at the first note). The source is the lane's In port, or the patch's MIDI input and surface keyboards when nothing is wired there. A take is one undo step.

Add from **+ → Sequencers → Drum / Melody / Chord / CC Lane** — Konstrukt creates the matching sequencer node and binds the control automatically. Long-press the control's edge and turn on **Hide Layer Picker** to collapse it back to a plain pattern row (the Prob/Tie/Cond data stays on the node either way).




### Scene Pads

`Binding: none · drives every bankable lane's Pattern`

A **scene launcher**: pads **A–D** (more as banks fill) that point every lane with banks — Drum, Melody, Chord and Analog lanes, Drum Grids, Matrices, Stage Seqs and Piano Rolls — at that bank in one tap. Each lane switches at its launch boundary, so the whole kit lands on the downbeat.

A scene never stops a lane: one with nothing in that bank switches to it and goes quiet, and a stopped lane starts again. The lit pad is the **armed** scene (every playing lane agrees); dots mark banks still sounding during the crossover. A named or coloured scene shows its name and colour. Lanes running a **Chain** keep cycling — set Chain to Off to hand them to scenes. Add from **+ → Performance → Scene Pads**.



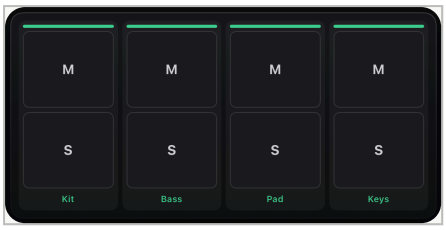
### Mixer

`Binding: none · shows the patch's mixer channels`

The **Rack page's mix on a card**: one strip per channel — level meter, fader, **M**ute and **S**olo, the channel's name and colour — with the **master meter** at the right. Put it beside your pads and Scene Pads and a whole performance fits one page; in the AUv3 it is the mixer.

Strips share the card's width and the card scrolls sideways once they would get narrower than a fingertip; on a short card the faders fold away and the meters, M and S remain. In the control's settings: **Master Meter**, **Send Buses** (their return faders join the strips), **Pan**, and **Channels** — All, or just the ones you pick (drums and bass, say).

The card plays the mix; building it — inserts, sends, routing, plugins — stays on the Rack page. Everything is the same state: a fader moved here has moved there, and **solo is global**, so it also silences channels the card doesn't show. Add from **+ → Performance → Mixer**.



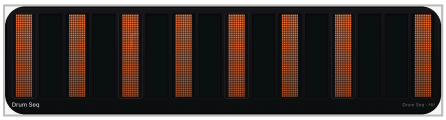

### Mutes & Solos

`Binding: none · shows the patch's mixer channels`

Just the mixer's **M**ute and **S**olo buttons: one tile per channel, big enough to play — drop the drums, solo the bass, bring them back on the one — without spending a page on faders and meters. Tiles wrap into rows to fill the card.

Each **mute group** (made on the Rack page) adds a tile whose button engages the group; a channel muted by a group shows its M half-lit. In the control's settings: **Send Buses**, **Mute Groups** and **Channels** — All, or just the ones you pick.

Same state as the Rack page and the Mixer card, and **solo is global**. Add from **+ → Performance → Mutes & Solos**.

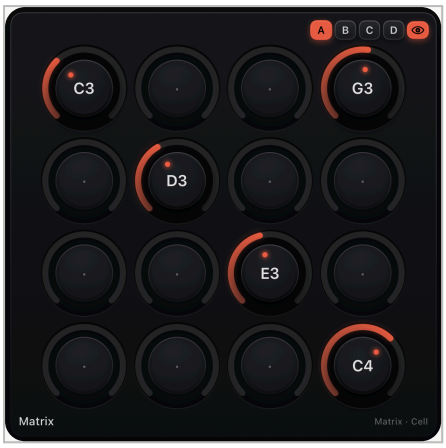


### Step Grid

`Binding: sequencer node ·` `hit1…16`, `note1…16` or `val1…16` `prefix`

16-step row with moving playhead. Tap toggles steps; drag across a row to draw a run. Long-press a step and drag horizontally for **microtiming** nudges. Melody lanes show the **sounding pitch** after the bound node's **Root** + **Scale** quantizer.

The plain, single-layer grid — new lanes get **Sequencer Lane+** above instead; use this (or Lane+ with **Hide Layer Picker**) when you just want the pattern row.




### Matrix Grid

`Binding: Matrix node · cell1…64 grid`

A grid of knobs (default 4×4, up to 8×8) mapped to a **Matrix** node's cells — each knob sets that cell's note (0 = rest), and a **live playhead** highlights the cell currently playing. Knob labels reflect the **sounding pitch** when **Root** + **Scale** quantize is on.

Auto-attached when you add a **Matrix** node; resize the control and the node's **Width** / **Height** follow.

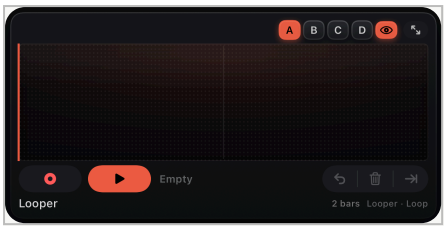


### Stage Panel

`Binding: Stage Seq node · pitch1…64 prefix`

The **Stage Seq**'s front panel, one column per stage: the stage number (tap to make it the **last stage**), a **pitch slider** (drag up/down; sweep sideways to paint several), the **pulse strip** (tap or drag to set 1–8 pulses — it draws where the gates fire), the **gate mode** (tap: Rest → Single → Repeat → Hold) and the **Slide · Skip · Ratchet** switches.

The playing stage lights, and so does the pulse inside it. The bar on top cycles the **Direction** and picks the **bank** (the eye follows the sounding one). Auto-attached when you add a **Stage Seq**.




### Piano Roll

`Target: Piano Roll node · Record / Play / Undo / Clear`

Transport for a **Piano Roll** node: record overdubs as takes, play the loop, undo the newest take, clear the buffer. Shows loop length, the quantize grid, the recorded phrase and the playhead, or (with **Edit on Card** on) the clip as an editable roll. The ⤢ button opens the full **editor**: Select / Draw / Erase tools, marquee selection, pinch-zoom, a velocity lane, quantize, legato, transpose, duplicate, copy / paste, loop length and ×2, with its own undo. **Send to lane** quantizes the take onto a Drum Seq or Step Seq lane's grid (its selected bank, replaced). Add from **+ → Sequencers → Piano Roll** to splice the node automatically.

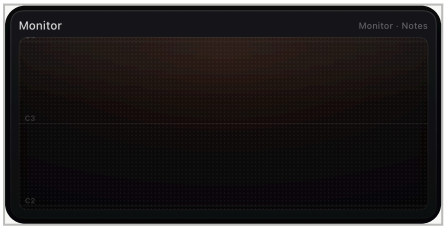
## Visualizers & debug — scopes and monitor




### Monitor

`Target: Monitor node · text log`

Scrolling text log of MIDI at a splice point. Add from **+ → Monitors → MIDI Monitor** — splices a **Monitor** node into the signal path.



### Note Scope

`Target: Monitor node · piano-roll view`

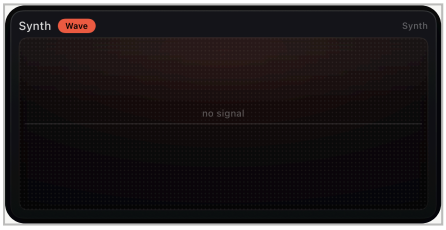
Scrolling piano-roll of note-ons and offs — see arps, sequencers and generators as they play.




### CC Scope

`Target: Monitor node · CC trace`

Oscilloscope-style CC history: value on Y, time scrolling. One color per CC number — watch filter automation and LFOs.




### Audio Scope

`Target: an audio node's output (audio) · none = the master`

Sound on a card: **Wave** is an oscilloscope (5–200 ms, held still by a rising-edge, free-run or **Beat** trigger — the window starts on the transport beat, so a kick pattern stands still at tempo), **Spectrum** an analyser on a log axis (20 Hz–20 kHz, dBFS, peak hold, the patch key's root ticked) and **Stereo** a goniometer with a correlation meter that turns red when the mix goes anti-phase for a beat. Add from **+ → Monitors → Audio Scope**.

The inspector's **Source** picks a channel (an instrument, Audio In or an effect's tail, tapped where its meter reads) or the **Master** (what the speakers get). A scope costs the engine nothing until it is on screen.

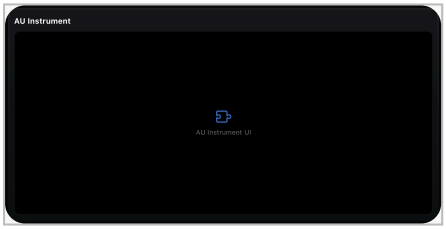



### Drum Grid

`Binding: Drum Grid node · hitA prefix`

The whole kit on one card: a **Drum Grid** node's eight rows — one drum sound each, named after its note — across 16 steps. Tap a cell to toggle a hit, drag to paint a run, long-press a cell and drag sideways to nudge its timing; the layer chips switch the grid to **Vel**, **Prob** and **Ratchet** (tap cycles, drag up/down dials) and **Cond** (tap cycles the trigger condition). Tap a row's name to mute it.

Everything a lane has: **A·B·C·D** banks with auto-follow, ◀ ▶ rotate (all rows together), the ● **record** chip (step record lands every pad you hit together on the cursor column; live record sorts each hit into its row by note) and a shared playhead. Add from **+ → Sequencers → Drum Grid**.




### Plugin View

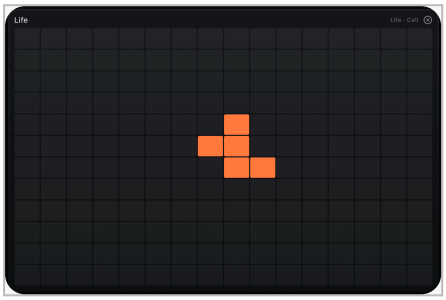
`Binding: AU Instrument / AU Effect / AU MIDI Effect node, or a built-in Analog, Drums, MegaSaw, Wavetable, Sampler, Slicer, Drive, Filter, Delay or Reverb`

A hosted AUv3 plugin's own UI as a card on the surface, so a synth's panel sits next to the knobs and lanes that drive it. The plugin fills the card; drag the card taller or wider and the plugin picks the nearest size it supports.

Bound to a built-in instrument or effect, the card shows that node's own panel instead — every parameter on one page, scaled to the card — and it can stay on the surface while the same panel is open in its floating window.

**Performance → Plugin View**, or **Show on Surface** in the node's inspector. Standalone app only.

## Emergence views — Life, Turing, scopes




### Life Grid

`Target: Life node · cell1…192 seed`

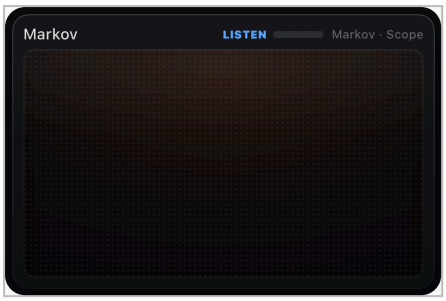
16×12 **Game of Life** colony. Tap cells to draw the seed; the pattern evolves while transport runs. Attached automatically when you add a Life node, or bind manually.




### Register

`Target: Turing or Rule node · live bit loop`

Live view of a **Turing** shift register or **Rule** automaton row — active steps, playhead and mutations as the pattern evolves. Tap or drag to poke **Rule** cells on or off.




**Scope**

`Target: emergence node · physics / melody trail`

Node-specific visualizer: Markov note trail, Bounce ball physics, Attractor trajectory. Pairs with **Markov**, **Bounce** and **Attractor** nodes.

## Factory preset list

**Boot patch: First Light** — generative Life colony + arp; loads on launch.

**Grooveboxes & lines** (self-playing): **Witching Hour** — horror synth in A harmonic minor: a Drum Grid on the Eighties kit whose clips (a heartbeat under a ticking clock, a half-time stalk, a chase) a dice LFO re-rolls every two bars, lifting the bass with them; a Stage Seq octave pulse through Am · F · Dm · E; a seven-step music box that slips against the bar through a tape delay the Lorenz attractor ages; a ghost choir on AmMaj7 · Fmaj7 · Dm7 · E7; bats chirping on the echo and wind over an A pedal; push the choir around with the **Spirits** pad, and the **Theremin** page is a scale keyboard in the key on a gliding, vibrato'd sine; **Night Drive** — synthwave in G minor: a Drum Grid on the Eighties kit through a gated room, three clips (breakdown, verse, chorus) the **Scenes** pads pick, a 1/16 octave bass on the PWM Bass, Gm7 · Ebmaj7 · Bbmaj7 · F7 on PWM Strings pumped by the kit with a two-bar LFO on its filter, the same changes arpeggiated up and down on the MegaSaw's Trance Pluck, a chorus-only lead through a tape drive; the **Drive** page is a scale keyboard on the lead; **Echo Chamber** — dub in E minor: a Drum Grid on the Reggae Kit through a **Phaser** (one drop, steppers, the drop-out), a round bass on the Moog Bass with slides, Em · Em · C · D skanked by the rhythm grid (two and four, the offbeat eighths on steppers), a melodica line that drops notes, laser zaps, and a dub desk on one page: two **Throw** buttons that fling the skank or the kit into the tape delay while held, a **Dub** pad over the delay's feedback and age, a **Siren** pad that plays its own synth into the echo, the mixer; **Groovebox** — three drum lanes and a sliding bass, each with two pattern banks flipped from **Scene Pads**; **Infinity Canon** — one fractal melody at three time scales at once, a self-similar prolation canon locked by the sequencers' Reset inputs; **Clockwork** — a solid 4/4 anchor with polymetric hats, clave and a 3-beat bass gearing against it, re-locked every N bars (or by hand from the Snap pad) through the sequencers' Reset inputs; **Amen Engine** — a generative jungle breakbeat machine: three drum lanes cut an Amen-style break that never plays the same bar twice (per-step probability, trigger-condition fills, Ratchet snare rolls, Humanize) at 170 BPM on the chopped Amen fill itself, with a scale-locked reese sliding underneath through Glide; **Berlin School** — a classic modular sequencer patch in three rows of eight knobs: a 1/8 riff with per-step accents and frequent 1/16 Ratchet stutters, transposed bar by bar through the Analog Seq's note latch by a slow Roots row whose EOC also re-locks the phrase, over a CC Seq stepping the filter and into a dotted-eighth analog echo; **Chord Voyage** — the Chord Seq front and center, in A minor: one lane carries three four-bar progressions the Chain cycles A·B·C (Am · F · C · G; Am ·

Dm · E7 · Am, the E7 borrowed from the major; F · G · Am held), every step voiced so the top note walks; the chords land on an evolving wavetable pad through a chorus into a hall, run through an Arpeggiator into a harp on the echo, with the roots walked by a Step Seq bass and a Linn kit ticking on a Drum Grid; the **Play** page is a chord keyboard into the pad and the arp; **Slide Bass** — the acid line on the Stage Seq, in C minor at 126: eight stages of one to eight 1/16 pulses, Hold stages for the long notes, Slide stages arriving legato through Mono + Glide, a Repeat stage hammering, a ratchet doubling a stab; three scenes (the line, the octave stabs, the breakdown of long slides) over a 909 on a Drum Grid; the Acid Bass through a tube Drive into a ladder Filter whose cutoff a sample-and-hold LFO re-rolls every sixteenth (**Step**), on a ping-pong echo; the **Play** page is a scale keyboard into the same slide chain.

**Emergence** (mostly self-playing): Conway Choir (a Life colony on a wavetable choir into a hall), Turing Bass (a mutating reese through a tape drive, a second machine rolling the hats over a kick), Cellular Drums (three Wolfram rules on the Drums' Minimal kit), Infinity Canon (three instruments for the three time scales), XY Matrix, Lorenz Skies and Markov Mirror (both playable from the surface: chord pads into the latched arpeggio, keys into the Markov mind), Trance Gate (chord pads into the gate on the MegaSaw).

**Also included:** Solo Lead (mono + glide portamento keyboard), Grand Keys (a house piano into a hall), Harmony Keys, Split Looper, Seaboard and Fourths Grid (MPE, pressure and slide routed into the voice with **Press →** and **Slide →** selectors), Drum Pads, the Mixer, MIDI Monitor, Scale Lock, Transposer, Tighten and CC Shaper — browse by category in the preset browser.

## Troubleshooting

- **No sound** — add an instrument node, turn on **Built-in Sound** in Settings, or route **MIDIRack Out** to a synth. In the AUv3, put a synth after the plugin.
- **Doubled sound** — instrument nodes and Built-in Sound both play. Turn Built-in Sound off.
- **Patch is silent** — follow the activity LEDs in the graph and splice a **Monitor** at the first dark node.
- **Knob does nothing** — read its caption. `CC 74` goes to your synth, not the patch; rebind it in the inspector.
- **Sequencer frozen** — press play, or start the host's transport.
- **Stuck note** — tap **Panic** (**⌘.**). If external gear still rings, it ignores All Notes Off; retrigger the note on the device.