USER MANUAL

Control your rig from your Stream Deck

MIDIRover is a free Stream Deck plugin that sends MIDI to **Neural DSP** and any MIDI-enabled plugin or DAW. This guide covers everything — installation, every action, Learning Mode, and a Neural DSP walkthrough.

Version 1.0.0.0 · Updated 11 September 2026 · Windows & macOS

---

1 What you need

- A **Stream Deck** — the key actions run on any model with keys (Mini, MK.2, XL, Neo, Stream Deck + and + XL, the Pedal), the dial actions on any model with dials (Stream Deck +, + XL).
- The **Stream Deck app** installed (Elgato, free).
- **Windows:** a virtual MIDI port — the free loopMIDI.**macOS:** nothing extra to *send* — MIDIRover creates its own port automatically. For the optional **Feedback** input it needs a two-way bus: switch on an **IAC Driver** bus in *Audio MIDI Setup*. MIDIRover's own port can be listened to, but nothing can send into it.
- The software you want to control — **Neural DSP** (any plugin), another amp sim, a synth, or a DAW that accepts MIDI.

2 Install & first run

1

Install the plugin

Add MIDIRover from the Elgato Marketplace. It appears in the Stream Deck app under the **MIDIRover** category.

2

Set up a MIDI port

Windows: open loopMIDI and create a port (the default “loopMIDI Port” is fine). **macOS:** skip — the port is automatic.

3

Pick the port

Drop any MIDIRover action on a key and open its settings. Choose your port under **MIDI Port** — or leave it on **Auto** (Windows: **Auto-detect (prefers loopMIDI)**; macOS: **Auto (virtual MIDIRover port)**). The choice is shared by every key, and the line just under the dropdown confirms where MIDI is actually going.

Point your software at the same port. In Neural DSP (or your DAW), set the MIDI *input* to the port MIDIRover is sending on — your loopMIDI port on Windows, or **MIDIRover** on macOS. Nothing crosses until both ends name the same port.

The line under the dropdown tells you what is happening. Every MIDIRover key shows a live port status directly beneath **MIDI Port**, and it always names the port. **Grey** — MIDI is going where you meant it to. **Amber** — MIDI is going out, but to a different port than this key is set to (on a Mac this is normally MIDIRover's own port standing in for a saved port that isn't there; select **MIDIRover** as the input in your music software). **Red** — nothing is being sent at all, and the line says why. When a key seems dead, read this line first.

3 Core concepts

MIDI port

The virtual cable between MIDIRover and your software. One port is shared across all your keys — set it once.

Channel

MIDI has 16 channels (1–16). Your key and your software must agree on the channel. Most setups leave everything on channel 1. New keys default to the channel you last picked, so once you move your rig to, say, channel 2, every key you add follows automatically.

CC, Program, Note

The three everyday message types. **CC** = a knob/switch (0–127). **Program** = recall a preset. **Note** = a musical note on/off.

MIDI Learn

Most plugins have a "MIDI Learn" mode: you arm a control, send it a message, and it binds. MIDIRover sends the message — you press the key to teach your plugin.

Colour theme

A shared **Deck theme** recolours every MIDIRover key and dial at once — pick one of six presets, or set your own two colours, in any action's settings. Each key can also override it: its **This key** setting follows the deck theme by default, or takes its own preset or custom colour, handy for colour-coding by function. The second box on both rows sets the key **background** — dark (the default) or light — deck-wide or per key; presets adjust their colours per background so both stay readable, and on light keys the key's Name is drawn in black as part of the face. (This reaches the **Stream Deck +** touch strip too, on every Display — **Bar**, **Knob** and **Level meter** all follow the deck theme and a per-key override, because MIDIRover draws all three itself.)

4 Actions reference

Drag any of these onto a key (or a dial, on a Stream Deck + / + XL). Give it a **Name** to label the key — or leave it empty, and the key names the message it sends: **CC22 ch1**, **PC22 ch1**, **C4 ch1**, or a Bank pair's id and channel like **A ch1**. If you have already named that CC, program or note in its dropdown, the key uses that name instead, so you never type it twice — on every key type, dials included. A **Bank** key is the one that needs a word of explanation: in its usual Program Change mode it does not sit on one preset, it *walks* a range, so the preset name appears on the key's **value** line as the counter reaches it (**Lead** instead of **3**) while the top line keeps the **A ch1** that tells one counter from another. In Control Change mode there *is* one controller, so its name goes on the name line with the id kept: **Drive A**. The box beside **Name** sets its font size in px — 10 to 30, 22 by default. MIDIRover draws the label into the key face itself, so it always matches the key's colours and its light or dark background. The Stream Deck app's own **Title** field still works and draws on top in whatever font and colour you choose — so use one or the other, or the label appears twice.

Control Change CC

The workhorse for stomp-style controls — drive, delay, reverb, boost. Toggle an effect on/off, send a fixed value, or make it momentary (on only while held).

Toggle · Icon On / off each press — the key shows its state. Default display: the power icon.

Toggle · Switch (vertical) The same on/off, drawn as a lever switch — up (on) / down (off).

Toggle · Switch (horizontal) The same on/off, drawn as a lever switch — right (on) / left (off).

Toggle · Play / pause The same on/off, drawn as transport — play (off) / pause (on).

Fixed The same value on every press.

Momentary On while held, off on release.

Sweep · Slider Hold the key to sweep the value — a wah or volume ride. Default display: a linear slider gauge.

Sweep · Knob The same hold-to-sweep, shown as a rotary knob with a value ring.

Sweep · Value / Gauge The same hold-to-sweep, drawn by the **Display** row: the number alone, with its name, or as a Bar or Arc gauge — the MIDI Readout faces, on a key that sends.

CC Number

The controller number you map in your plugin (0–127). Controllers with a fixed job in MIDI are labelled for you — `122 · Local Control`, `123 · All Notes Off` — and you can **rename** any of them (`22 · Gain`), which shows on every key that offers a CC list for that channel — the MIDI Readout key's Number list included, though renaming itself lives on the keys that send (see section 6). Every other number stays bare on purpose — and not because it has no conventional meaning. CC 1, 7, 11 and 64 are widely used for Modulation, Volume, Expression and Sustain, but nothing stops a plugin repurposing them, and a label that is right most of the time cannot be switched off. What *is* pre-

	labelled are the controllers that carry a **function** in MIDI rather than naming a destination for a value: the channel-mode commands (`120`–`127`), the RPN/NRPN parameter mechanism (`6`, `38`, `96`–`101`), and Bank Select (`0`, `32`), which changes how the next Program Change is read.
Value On / Off	The values sent for the on and off states (default 127 / 0).
Latch group	Toggle keys sharing a group name act like radio buttons — turning one on turns the previous one off. Leave blank for independent toggles.
Sweep time / Range	Sweep mode: seconds for a full sweep (1–10) and the min–max span; the sweep bounces at the ends and remembers its position.
Fade	**Fade to value** mode: where a press heads (default **Reverse each press**). **Reverse each press** alternates — a press ramps to whichever end of **Range** is farther off, and the next press brings it back, so one key plays both halves of a swell. **Out** and **In** always head for the same end of Range. **To a set value** ramps to the number in **Fade to** and stops there — that row appears for this choice only, because the other three work the target out from Range themselves.
Fade to / Fade time	**Fade to value** mode: with **Fade** set to **To a set value**, the value the ramp travels TO — and, whichever direction is set, how long the ramp takes. One press and the value moves on its own — both hands stay on the instrument, which is what separates it from **Sweep** (where you hold the key for the whole move). Press again to stop it where it is; press once more and it carries on from there. Editing either setting mid-fade stops it too, since the ramp in flight was planned from the old numbers. Range clamps the target, so a target of 127 on a key ranged 40–90 ends at 90. Best on a continuous parameter — volume, mix, gain — because a switch-like one just flips somewhere in the middle.
Key graphic	**Toggle** mode: the **Icon** (power symbol), a lever **Switch** — whose axis you pick in the **Orientation** row below — or **Play / pause** transport (play when off, pause when on). **Sweep**

	mode: **Slider** draws a live linear gauge and takes the same **Orientation** row, **Knob** a rotary knob with a value ring, or **Value / Gauge** hands the face over to the **Display** row below. (Fixed and Momentary each show their own icon.)
Orientation	Which way the graphic runs, **vertical** or **horizontal**. Appears under **Key graphic** for the two graphics that have an axis: the **Slider** in Sweep mode (the Knob is always round) and the **Switch** in Toggle mode. A vertical switch flips up for on and down for off; a horizontal one flips right for on and left for off.
Format	Sweep mode only. Display only — it changes the text drawn in the value slot and nothing else: not the byte on the wire, not the stored setting, and not how often the key repaints. **Raw** (default) shows the MIDI value 0–127. **Percent** is relative to this key's own **Range**, so a key limited to 20–100 reads 100% at 100 — the number agrees with the full-looking gauge beside it. **Pan** reads L / C / R from the MIDI centre (64), which is absolute: a restricted range honestly never reaches L50 or R50 rather than pretending to. **Approx. dB** reads the value as a fraction of full scale (0 dB at 127, about −6 dB at half, **-inf** at 0) — called *approximate* on purpose, because without knowing your device's own taper no single curve is right for every DAW.
Display	Sweep mode, **Value / Gauge** graphic only: **Value only**, **Value + Name**, **Value + Bar** or **Value + Arc**. These four are shared with the **MIDI Readout** action, so a Sweep key and a Readout set to the same CC, channel and Range show the same picture — one sends it, the other watches it. (The Readout and the SD+ dial both offer a **Level meter**, which is not here: it exists for watching a signal arrive, and a key that sends already knows what it sent.) Bar and Arc fill across the **Range**, so give both keys the same Range or they will disagree.
Feedback	Optional MIDI input, shared by every action. For this key, an incoming CC keeps a Toggle's state and light or a Sweep key's value (and its on-key graphic — slider or knob) in sync with the MIDI your DAW sends back. Neural DSP is receive-

	only, so the source is your DAW's automation (or another controller), not the plugin itself. One virtual port is enough: MIDIRover recognises and discards its own output when a loopback port returns it, so the Feedback input can safely be the port it transmits on. Off by default.
Channel	MIDI channel 1–16.

Program Change PC

Jump straight to a saved preset or tone. Assign one program number per key — the fastest way to switch Neural DSP presets. (In Neural DSP, add a **Program Change** → **Preset** mapping for each preset first. In a DAW the host must forward Program Change — Ableton Live doesn't and Logic often doesn't, so the standalone app is the reliable path.)

Program	The program number to recall (0–127).
Bank	Bank Select sent right before the program (coarse CC0 / fine CC32) — for gear whose presets live in banks. Optional to *send*: leave on "–" to skip it and transmit the Program Change alone.
Numbering	**Display only** — whether the Program dropdown counts from **0** (as MIDI sends) or from **1**. Most hardware prints presets one higher than the byte, so pick 1-based if your device shows one more than you sent. The MIDI byte is unchanged.
Channel	MIDI channel 1–16.
Feedback	Optional MIDI input, shared by every action. For this key it drives an **active-preset** indicator: the card drawn on the key **fills in** when its program is the one currently active, and empties when another preset takes over — so a row of preset keys shows at a glance where your rig actually is. A press fills the key too, which means the indicator still works with **no input selected**: there it shows the last preset MIDIRover sent on that channel — including one sent by a **Bank Up** or **Bank Down** key, so stepping through presets with a Bank pair moves the filled card along the row with it. Nothing is filled until a preset is chosen, because before that the current preset is genuinely unknown.

Bank Up / Down

PC / CC STEPPER

Classic preset scrolling, or in Control Change mode step a CC-Absolute value up/down (with wrap). Pair an **Up** key and a **Down** key with the same Bank id and channel — they share one counter and both show the current value.

Output	**Program Change** (default) sends the classic preset PC. **Control Change** sends a CC whose value is the stepped counter (0–127) — cycle or nudge a CC-Absolute parameter. A Dec/Inc parameter instead needs a Fixed-value key on each of its inc/dec CCs.
CC Number	Control Change output only: the controller sent (0–127). A CC Up/Down pair must share it (as well as Bank id + channel) to move one counter.
Direction	Whether this key steps up (next) or down (previous).
From / To	The ends of the range the counter walks. In **Program Change** mode these are preset numbers, so they show any **names** you have given those programs (`3 · Lead`) — the same names the key face prints. In Control Change mode they are values 0–127 rather than presets, so they stay plain numbers. Renaming a preset is done on a **Program Change** key, where you address one; here you are choosing where the counter stops.
Bank id	Keys sharing this id + channel move the same counter. Use different ids for independent banks. It also shows on the key (`A ch1`) when you have not typed a Name, because it is what tells two independent counters apart — so a Program Change bank keeps showing it even once you have named the presets, and the preset name appears on the value line instead. In Control Change mode a named controller reads `Drive A`, name and id together, for the same reason.
From / To	The range to move within (e.g. 0–9) — program numbers in PC mode, CC values in CC mode.
Step	How many to move per press.

At ends	**Wrap around** loops past the end back to the start; **Stop at ends** clamps.
Key shows	Show the current value on the key, or the name only.
Steps	Optional ordered list (`value:label, …`) replacing From/To — the key shows each step's label; one list item per press (the Step size is ignored in Steps mode, but At ends still wraps/clamps at the list ends). Paste the same list on the paired key. Items the plugin would skip outline the field in red as you type.
Feedback	Optional shared MIDI input — with one set, the counter **follows the outside world**. Change the patch at the gear (or in your DAW) and an incoming Program Change on this channel updates the key's number without it being pressed, so the next press continues from where the device actually is instead of from where MIDIRover last left it. In Control Change mode it follows an incoming CC matching this key's CC number instead. Off by default; without it the counter only knows what it last sent.

Note NOTE

Send a MIDI note — for parameters that learn to notes, tap-tempo, or triggering things in your DAW.

Tap Note-on then note-off on each press.

Hold Sounds while the key is held.

Toggle On / off per press — lights while the note is on.

Note On only Lets the note sustain — end it with a Tap on the same note.

Note	The note number (0–127), shown with its musical name — `60 · C4` by default. You can **rename** any note (handy for drums: `36 · Kick`), and your names appear on every key that offers a Note list — the MIDI Readout key's Number list included, though renaming itself lives on the keys that send (see section 6).
Naming	Which octave numbering the note names use — three choices. `Middle C = C4` matches Roland gear and most DAWs; Yamaha calls the same note `C3`; `C5` suits a few older sequencers. Display only — the note number sent never changes. This one setting applies to every key, and it also decides how an **unnamed** Note key labels itself: the same note reads `C4 ch1` or `C3 ch1` depending on this choice.
Velocity / Off velocity	How hard the note is struck (0–127), and the release velocity (default 0 — most gear ignores it). **Velocity 0 is a Note Off** by MIDI convention, not a very quiet note, so a **Toggle** key set to it neither lights nor latches: the byte still goes out exactly as configured, and the key does not claim to be holding a note that is not sounding.
Latch group	**Toggle** mode only. Give several Toggle keys the same group name and they behave as one switch, like a keyswitch set: pressing one sends **Note Off** for whichever member was on before sending its own **Note On**, so only one sounds at a time. Leave blank for independent toggles. Members may sit

	on different channels and ports — each is released on the channel and port it claimed, not the new key's. A member that was switched on by a previous run of the plugin is cleared without sending anything, because the note it names is no longer this session's to end.
Feedback	Optional shared MIDI input — an incoming note-on / note-off matching a **Toggle** action's note and channel keeps its state and key light in sync (e.g. when your DAW plays the same note). Off by default.
Channel	MIDI channel 1–16.

Advanced MIDI ADVANCED

For gear that needs more than CC / PC / Note. Pick one message type per key.

Pitch Bend 0–16383 (8192 = centre, 16383 = full up, 0 = full down). Sent once per press — the bend stays there, so pair a second key at 8192 to snap back to centre.

Aftertouch Channel pressure, 0–127.

NRPN 14-bit parameter + value (sent as CC 99 / 98 / 6 / 38).

SysEx Raw hex bytes — `F0`/`F7` are added automatically if you omit them. Separate bytes with spaces, commas, periods, colons, semicolons, or dashes; anything the plugin can't parse outlines the field in red and nothing is sent.

Transport Start, Continue or Stop — the System Real-Time commands that drive anything slaved to external MIDI clock: a hardware sequencer, groovebox or drum machine, or the Run/Reset gates on a MIDI-to-CV converter. **Start** rewinds and plays from the beginning; **Continue** resumes from where **Stop** left off. Each press sends a single byte (`FA` / `FB` / `FC`) and carries **no channel**, so the Channel row disappears in this mode. **Stop** sends only Stop — it does not silence held notes; use the Panic action for that.

Command — Transport mode only: which System Real-Time command this key sends — **Start**, **Continue** or **Stop**. Each is a complete one-byte message, so this key has no Channel.

MIDI Dial

MODELS WITH DIALS (STREAM DECK + / + XL)

Ride a parameter by hand — gain, mix, wah — on a rotary dial. The touch strip shows the name and the live value as a bar or a rotary knob. (The dial actions appear on any model with dials; the Stream Deck + and + XL are the two Elgato documents with a touch strip to display the value on.)

CC Number	The controller the dial sweeps (0–127).
CC mode	**Absolute** (default) sends the 0–127 value the dial rides. **Relative** sends +/− deltas for an endless encoder, so a DAW parameter never jumps on pickup — Range / Reset / Display and feedback don't apply, and the touch strip reports what the control IS and what it just SENT rather than a position it cannot know: the name (or the address), what one detent sends, and which **Encoding** is configured — then the moment you turn it, the signed amount that went out, with the direction lit. It settles back after about a second. There is deliberately no value and no gauge: a relative dial has none, and a number with a scale behind it would imply a range MIDIRover cannot see.
Encoding	Relative mode only: **Signed bit**, **Two's complement**, or **Binary offset** — match your DAW's relative-CC mode (Ableton / Reaper / Bitwig / Cubase all list these).
Step / tick	How much the value changes per detent. Hold the dial while turning for coarse steps (×4).
Reset to	Press and release the dial (without turning) or tap the screen to jump to this value (clamped into Range). Holding while turning gives coarse steps and never resets.
Range	Limit the sweep to a min–max span (default 0–127) — e.g. keep a wah between 20 and 100. The touch strip display follows the range.
Format	Absolute mode only — a relative dial has no absolute value to describe. Display only — it changes the text drawn in the value slot and nothing else: not the byte on the wire, not the stored setting, and not how often the key repaints. **Raw**

	(default) shows the MIDI value 0–127. **Percent** is relative to this key's own **Range**, so a key limited to 20–100 reads 100% at 100 — the number agrees with the full-looking gauge beside it. **Pan** reads L / C / R from the MIDI centre (64), which is absolute: a restricted range honestly never reaches L50 or R50 rather than pretending to. **Approx. dB** reads the value as a fraction of full scale (0 dB at 127, about −6 dB at half, **-inf** at 0) — called *approximate* on purpose, because without knowing your device's own taper no single curve is right for every DAW.
Display	**Bar** (default), **Knob** (a rotary knob — where its ring starts is the **Fill from** row below), or **Level meter**. **Level meter** draws twelve segments across the **Range**, fed by incoming MIDI — the same segments, and the same twelve steps, as the **MIDI Readout**'s meter face, so a dial and a key watching one CC always light the same count. It shows **no number**, because it repaints only when the lit count changes and a number would sit a moment behind the signal; **Bar** and **Knob** are the displays that print a value. Turning the dial still sends as usual. Not available in **Relative** CC mode, which does not follow feedback at all. Unlike the key's meter, a **MIDI Dial** does not show clipping: a dial stores the value it is sitting on, and that value is held inside the Range so that turning it continues from where it is — so by the time the face is drawn, a value above the Range has already become the top of it. That is a property of *this action*, not of the hardware: a **MIDI Readout** meter stores nothing and clamps nothing, so it *does* show clipping — on a key or on a dial. Put one on the same CC if you need to see it.
Fill from	**Knob** only. **Minimum** (default) fills the ring from the low end of the **Range** — what you want for a volume, a gain, a wah. **Centre** fills it out from the middle instead, right of twelve o'clock above the midpoint and left below, which is how a **pan** or any other bipolar control reads naturally. The pointer tracks the value either way; only the lit part of the ring changes.

	This was two entries in the **Display** list until it became its own row, because one dropdown was answering two different questions — which face to draw, and where its ring begins. Keys made before the change keep the look they had.
Feedback	Optional shared MIDI input — an incoming CC matching this dial moves its value and touch strip display (e.g. when the DAW automates the parameter). Off by default. (Not used in Relative mode.)
Channel	MIDI channel 1–16.

Panic ALL NOTES OFF

One press clears a stuck note or hung sustain — it sends All Sound Off (CC120), All Notes Off (CC123) and Reset All Controllers (CC121). Especially handy next to Note keys set to **Note On only** / **Hold** / sustain, which don't release on their own.

Send on	**All channels** (default, all 16), a single channel 1–16, or **Selected channels…** to list your own.
Channels	Only with **Selected channels…**: the channels to reset, as numbers and ranges — `2, 5, 9-12`. The line underneath names exactly what will be reset as you type. Use it to spare a channel that must keep running, like a looper. If nothing in the box can be read as a channel, Panic resets **all 16** rather than nothing — the whole point of the key is that it works when you need it. Numbers outside 1–16 are ignored, not rounded, so a mistyped `0` or `17` never resets a channel you did not ask for.
MIDI Port	Where the reset is sent. Panic reaches **one port** — this key's own, or the shared port when it is left on **Auto-detect**. A note sounding on a *different* port is not silenced by it, because MIDIRover holds one output port at a time: opening this one closes that one. With every key on Auto-detect or a single shared port — the default, and much the commonest setup — one Panic key covers the whole deck. If you deliberately split keys across two ports, give Panic the port whose notes you need it to stop, or place a second Panic key set to the other one.

MIDI Readout DISPLAY ONLY

A key that only **shows** a value — it never sends MIDI. Point **Watch** at the message you want to see and the key displays it live: a patch number, a fader position, a meter. It needs a **Feedback** port, and software that actually *sends* MIDI — Neural DSP plugins are receive-only, so use your DAW's automation or another controller. Until the first message arrives the key shows what it is waiting for (e.g. `CC22 ch2`), and it says **Feedback off** if no input is selected — so a blank-looking key always tells you which of the two to fix. There is a third message: if the port you picked later *disappears* (loopMIDI closed, an interface unplugged, the DAW quit) the key says **Port lost**, which is deliberately not the same words — one asks you to choose a port, the other to bring one back. When the port returns the key picks up again by itself. Note what is *not* a problem: a face that keeps showing its last value while nothing arrives is working correctly. A Control Change is a position rather than a stream, so a fader that has stopped moving sends nothing and the last value is still the truth. The **Peak hold** marker is the deliberate exception — it measures how high the signal reached RECENTLY, which does age, so it falls on its own while the value stays put. One more case that looks like a fault and is not: **restarting the plugin clears every reading**. If the Stream Deck app restarts, or the plugin updates or recovers from a crash, each watching key goes back to a dash until its next message — it has genuinely heard nothing yet, and a Control Change sender does not resend on its own. Move the fader (or send anything on the address) and the key fills again. A small `RX` in the top-left corner marks the key as a watcher rather than a sender, because the sending keys name themselves the same way — without it a Readout and a Control Change key on the same address would look identical.

It also runs on a **Stream Deck + dial**, drawn on the 200×100 touch strip instead of a key. There **Display** offers four faces rather than five — **Value only**, **Value + Name**, **Level meter** and **Needle** — because **Value + Bar** and **Value + Arc** have no dial form, while **Needle** is the one face a key cannot draw; your choice is kept if you move the action between a key and a dial. The **Name** font-size box is hidden there, since the dial faces draw their title at a fixed size. **Pushing or touching** the dial clears the peak marker and does nothing else: a Readout never transmits, so there is no value to adjust. With **Peak hold** off there is no peak to clear, and the on-device gesture hints say so rather than advertising a gesture that cannot happen. Watching **Program Change** the bottom line stays `PC ch1` even after messages arrive, and that is deliberate: the key is not tied to one preset, so what it watches is the channel, and the preset that arrived — its **name**, if you have given it one — is what fills the big slot.

Watch	**Control Change**, **Note velocity** or **Program Change** — the kind of message to display.
Channel	MIDI channel 1–16. Must match the sender.
CC / Note number	Which controller or note to watch (hidden for Program Change, which carries its own number). Until you pick one, it follows **Watch**: **22** for Control Change and **60** for Note velocity, matching the default a Control Change key and a Note key already use. Once you pick a number it stays put, whichever way you switch Watch afterwards.
Naming	Note velocity only, display only: which octave numbering the note names use. Shares the one setting with the MIDI Note key, so the two panels always agree.
Numbering	Program Change only, display only: show programs as **0–127** as sent, or **1–128** to match gear that counts from 1. The **Range** ends count the same way, so a gauge never disagrees with the value above it. If you have **named** that program (in any Program Change key's dropdown, or from an imported name map), the key shows the **name** instead of either number — Lead rather than 3 — and long names are shortened with an ellipsis to fit the key. The name is attached to the program as MIDI sends it, so it is the same name whichever numbering you pick.
Format	Display only — it changes the text drawn in the value slot and nothing else: not the byte on the wire, not the stored setting, and not how often the key repaints. **Raw** (default) shows the MIDI value 0–127. **Percent** is relative to this key's own **Range**, so a key limited to 20–100 reads 100% at 100 — the number agrees with the full-looking gauge beside it. **Pan** reads L / C / R from the MIDI centre (64), which is absolute: a restricted range honestly never reaches L50 or R50 rather than pretending to. **Approx. dB** reads the value as a fraction of full scale (0 dB at 127, about −6 dB at half, **-inf** at 0). It is called *approximate* on purpose: without knowing your device's own taper no single curve can be right for every DAW, and MIDIRover would rather show one honest

	approximation than ship a per-DAW curve file for you to maintain. **Pan** and **Approx. dB** are offered for **Control Change** only: a note readout's value slot shows the note's **velocity**, which is a magnitude — so **Percent** suits it — but it is neither a position nor an amplitude, and a note-off (velocity 0) would otherwise read **L50** or **-inf**.
Display	**Value only**, **Value + Name**, **Value + Bar**, **Value + Arc** or **Level meter**. The first four are the same faces a Control Change key in **Sweep · Value / Gauge** can draw, so a Readout and the key driving the same CC can show the same picture. Watching **Program Change** offers only **Value only** and **Value + Name**: a program number names a patch rather than measuring anything, so a bar, an arc or a meter would draw program 64 as a fraction of a range it does not sit on. On a **dial** the choice is **Value only**, **Value + Name**, **Level meter** or **Needle** — Bar and Arc have no dial form, and **Needle** has no key form, because its sweep needs the strip's width. A **Needle** draws a pointer over an arc that fills as the value rises: a held peak shows as a dot on the scale, and a value above the top of **Range** turns the last slice of the arc red. It is deliberately not called a VU — a VU is a defined instrument with a dB-scaled scale and a calibrated reference, and this is the same linear reading the **Level meter** shows, in the shape of an analogue movement. **Level meter** is a twelve-segment bar fed by whatever arrives. It is on this action and on the **SD+ dial** — both are for watching a signal move — and not on the sending keys, which already know what they sent. It shows **no number**, on purpose: it repaints only when the lit segment count changes, so a number beside it would sit a moment behind the signal. Use **Value + Bar** when you want the exact figure. The mapping is **linear across the Range** and the meter is **not calibrated** — it is a level indicator, not a VU: there is no dB curve and no per-DAW correction, because without knowing your device's own taper no single curve is right. If the value goes **above** the top of the Range, the **top segment turns red** — the meter has run out of scale and is

	no longer telling you how loud it is. At exactly the top it stays accent-coloured, because pinning there is the meter working rather than failing. The warning **holds for about two seconds** after the last over-range value, because a clip is an *event*: a single message above the Range would otherwise be gone before the key next repaints. It is shown whatever **Peak hold** is set to — that setting governs the falling marker and nothing else. This is the one case a full meter cannot otherwise show: with a Range of 40–90, every value from 90 to 127 looks identical without it. A meter is cheap on the deck's repaint budget but not free, so one or two per profile is comfortable; a deck full of them will compete with each other. Every face is available for **Control Change** and for **Note**. A Readout watching notes shows the **velocity**, which is a magnitude, so a gauge of it reads as fairly as the percentage does — and a **Level meter** with **Peak hold** is one of the better uses of this action, since the marker keeps the hardest recent hit that a bare number cannot show. **Program Change** is the one exception, for the reason given above: a program names a patch rather than measuring anything. That exception is enforced by the plugin and not only by this panel, so it holds for a key you copied from another profile as well as one you set up here. If a key showing a **Level meter** is switched to **Program Change** it draws **Value + Name** instead — and your choice is not thrown away: switch **Watch** back to **Control Change** and the meter returns with nothing to retype.
Peak hold	**Level meter** only. **Off** by default. It controls the falling marker and *nothing else* — the red clip warning is shown either way. **Mark the highest segment** keeps a bright marker at the loudest segment reached and lets it fall back down the full scale in about twelve seconds — a segment a second on this face's twelve — so you can see a peak you would otherwise have missed between glances. The *time* is what is fixed, so a meter here and a bridge column drawing

	eight bars empty together. It costs a few extra repaints, because the marker can move on a message the level alone would have skipped — measured at about 2.2 a second against 1.6 without it, both far below the deck's ten. It falls whether or not values keep arriving: when the signal *stops* mid-fall the marker keeps stepping down once a second until it lands, then the key goes quiet again. That tail is bounded by the marker itself — at most twelve repaints for one falling from the top, and none once it has landed — so a key watching a dead source settles to an honest empty face instead of holding a line that looks live.
Range	The low and high ends of the **Bar** and **Arc** gauges, the span the twelve segments of the **Level meter** cover, and the span **Percent** is measured against — so it appears for any of those three Displays, or whenever Format is Percent (0–127 by default). Values outside it pin to the end. On a **Level meter** the Range *is* the scale. Left at 0–127 a signal that never exceeds 40 lights only the first four segments; set the Range to 0–40 and the same signal uses the whole meter. With Format set to Raw (the default) the number is the value received; the other formats describe that same value differently.
Feedback	The shared MIDI input to listen on — one setting for every MIDIRover key. This action does nothing until it is set.

MIDI Meters

DISPLAY ONLY, DIALS

A **meter bridge** on one Stream Deck + dial: up to four level meters side by side, each watching its own address. It never sends MIDI, and like the Readout it needs a **Feedback** port and software that actually *sends* MIDI. **1** draws the wide horizontal meter this plugin already uses; **2** to **4** draw vertical columns, and the column count decides their resolution — a vertical column gets **8** segments where the horizontal one gets twelve, because at four columns twelve segments are under 4px each and read as a texture rather than as steps. Each column names itself with the address it watches until you name it; from three columns up the channel is dropped from that label, because a 54.7px column shrinks `CC22  ch1` to 11px and cannot fit `CC127 ch16` at all.

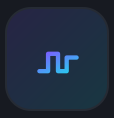
The panel gives each column a **tab** carrying its own full set of settings — name, what it watches, channel, number, range and peak hold — so four columns are configured the same way four separate keys would be, one at a time rather than in a wall of rows.

Every column holds its own **peak marker** and its own **clip warning**, on the same rules as the single Level meter: the marker falls the full column in about twelve seconds, and the red top segment holds for about two seconds after a value above the Range. **Pushing or touching** the dial clears every column's marker, and does nothing else — with no marker held, the deck says so rather than advertising a gesture that would do nothing. With no **Feedback** port set the strip says **Feedback off** in words, because four empty columns are indistinguishable from four live signals resting at zero.

Feedback	The shared MIDI input to listen on — one setting for every MIDIRover key. This action does nothing until it is set.
Meters	How many columns the strip draws: **1** (the wide horizontal meter), or **2**, **3** or **4** vertical columns. It also decides how many tabs there are — setting it to 2 hides the Meter 3 and Meter 4 tabs, and their settings are kept, not cleared.
Display	Whether the columns are labelled. **Meters + Name** draws each column's name above it; **Meters only** hides all of them at once and gives the bars the freed height — a segment grows from 6.75px to 8.75, so a four-up bridge gets about a

	third more meter. Useful once you know which column is which. It applies to the whole strip, not per column, so it sits outside the tabs.
Name	What this column is called, drawn above it. Leave it empty and the column names itself with the address it watches, or with whatever you have renamed that CC or note to. At four columns a long name is shortened to fit; an address never is.
Watch	Whether this column follows a **Control Change** or a **Note velocity**. One bridge can mix the two — each tab chooses for itself.
Channel	The MIDI channel this column listens on, 1–16, and a box to name that channel. Channel names are shared with every other MIDIRover key.
CC number	Which CC this column follows (or which note, when **Watch** is Note velocity — the row is labelled **Note** then). Defaults to CC 22 or note 60, the same defaults the Control Change and Note keys use, so a column dropped beside one watches what it sends.
Range	The low and high values this column's segments span, per column — eight of them when the strip shows two to four meters, and twelve when it shows one, because a single meter is drawn as the same wide face the Readout's **Level meter** uses. A value above the high end lights the top segment red — which means the default 0–127 can never clip, since no CC can exceed 127; narrow the range to see it.
Peak hold	Marks the highest segment this column has reached and lets it fall back down the full column in about twelve seconds, per column — the same time a Readout's meter takes, so a bridge and a Readout watching one CC agree about it despite drawing eight bars against twelve when 2–4 are shown. Set to **1** the bridge draws the wide twelve-segment face instead — the same one the Readout's **Level meter** uses — so there the two are not merely comparable but identical, and the marker steps in twelfths. It keeps falling when the signal stops, so a column whose source has gone

quiet empties rather than holding a line that looks live — a column falling from the top costs at most one repaint per segment, and the strip is then silent. It does not affect the clip warning, which is always on. Push or touch the dial to clear every column's marker at once — rotating deliberately does nothing, so a knob turned by accident cannot wipe a reading.



MIDI Monitor DISPLAY ONLY

A key that shows the **last MIDI message that arrived** — its type, channel, number, value, and the raw bytes underneath. It never sends anything. This is the key to reach for when something is not working: it answers *did anything arrive at all*, which nothing else on the deck can tell you. It needs a **Feedback** port, and says **Feedback off** until one is picked, or **Waiting for MIDI** once a port is set but nothing has spoken yet — so a blank-looking key always tells you which of the two to fix.

Channel	Leave on **All** to see everything. Pick one channel and the key ignores the others — and also ignores Clock, Start/Stop and SysEx, which carry no channel at all.
Show	**Musical messages** (the default) hides the housekeeping traffic that never stops: Timing Clock, Active Sensing and MTC. Clock alone is 24 pulses per beat — about **48 messages a second** at 120 BPM — so with **Everything, incl. clock** selected the key shows little except **Clock**. Switch it on to confirm a clock is arriving, then switch it back.
Feedback	The shared MIDI input to listen on — one setting for every MIDIRover key. This action does nothing until it is set.

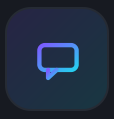
Tempo

DISPLAY ONLY

A key that **reads the tempo** of whatever is sending MIDI clock, and beats along with it — once a beat, so counting it gives you the tempo — with the measured BPM above and the key's name below. It never sends anything. Pick how it keeps time in the **Display** row: a **metronome** rod that swings, or a **flash** that pulses the key in your accent colour — one to look at, one to catch in the corner of your eye. **Something has to be sending clock:** a DAW will if you switch on clock output for the port MIDIRover listens to, while a standalone amp-sim plugin will not send it at all, so this key does nothing for that setup. Until clock arrives it reads **No clock**, and **Feedback off** if no input is picked — so a key that is not moving always tells you which of the two to fix. Stop dims the rod, or stops the flash, so a paused DAW still shows its tempo without pretending to play.

Display	How the key keeps time. **Metronome** (default) draws a rod hinged inside a wedge, swinging left and right on each half of the beat — something to look at directly. **Flash** replaces it: the **whole key** lights up in your key colour once a beat — lit for the first quarter of the beat and dark for the rest — so you can read the beat without looking straight at the deck. With no rod in the way, Flash also draws the BPM **large in the middle** of the key, exactly as a Control Change key set to *Value only* draws its value; the key's name stays in the usual place below. The flash uses the palette's own two colours on *both* key backgrounds — elsewhere a light background gets a darkened variant of the accent, which is right for thin lines but made a bright preset like **Emerald → Lime** read as dark green and olive across a whole key — and on the lit beat the text switches to whichever of black or white reads better against your colours — black for all six presets, and white if you pick a custom pair dark enough to need it. Neither display flashes while the transport is stopped.
Average over	How many beats the reading is measured across: **1 beat**, **2 beats** or **4 beats**. This changes how the number *looks*, not how right it is — one beat is already accurate to a fiftieth of a BPM. Pick 1 beat to see a tempo change the instant it happens (about 0.7 s), or 4 beats for a number that sits still

	(about 2.6 s to catch up). At 1 beat the last digit may flicker between neighbouring values; at 4 beats it holds.
Feedback	The shared MIDI input to listen on — one setting for every MIDIRover key. This action does nothing until it is set.



Label TEXT ONLY

A key that only shows **text**. It sends no MIDI, listens for none, and a press does nothing — use it to annotate the deck: a heading over a row of stomps, or a note beside the key it explains. The Stream Deck app can put a title on any key, but every other action *does* something when pressed, so a label built from one is a trap for whoever touches the deck next.

Text	What the key shows. It wraps to fit, and a line break you type is kept. There is no separate Name row on this action: elsewhere the Name is a caption under a graphic, here the text *is* the key.
Size	**Auto** picks the largest size that shows everything *and* keeps every word whole, so a short word comes out big and a sentence smaller. Pick a fixed size (10–30 px) to match neighbouring labels instead; text too long for it is cut with an ellipsis rather than overflowing.
Align	Two pickers, one decision: vertical (**Top** / **Middle** / **Bottom**) and horizontal (**Left** / **Center** / **Right**), so a caption can sit against the edge nearest whatever it describes.
Text colour	**Default** follows the light or dark key background — that is what keeps a Label looking like a label rather than a lit control. **Accent** paints the text in the **Deck theme** colour instead, which is useful for a heading above a row, and is the only setting on this action that the theme's *colour* affects at all (the light/dark background always applies). Small text uses a solid accent rather than the gradient, which turns muddy at that size.

5 Learning Mode

The fast way to set up a key — no typing CC numbers. Available on **Control Change**, **Note**, and **MIDI Dial**.

1

Open the Learn row

In the key's settings, find **Learn**. Pick the MIDI input your controller is on — or leave **Auto (first input)**.

2

Press Learn

The key reads **Listening…**. Move a knob, press a switch, or turn a dial on your controller.

3

It fills itself

MIDIRover captures the CC, Note or Program Change and fills the Channel and number for you. Status shows what it learned.

Listening stops after about **12 seconds** if nothing arrives, and you can click **Learn** again to cancel. The input port is only opened while you're learning.

6 Naming values

Give channels, CCs, programs and notes friendly names so the dropdowns read `22 · Drive` instead of just `22`.

- Type a name in the small field next to a Channel / CC / Program / Note dropdown.
- Names are **shared across every key that offers that dropdown**, so you label a value once — and **every list of channels, CC numbers, programs or notes shows them**, including the three you cannot type a name *into*. Those three are read-only for three different reasons. The **MIDI Readout** key's Number list is the same field for a CC number or a note number depending on what the key is watching, so there is no one list to rename into — it reads both, following **Watch**. The **Panic** key's **Send on** list is a target selector with an **All channels** option rather than a plain channel dropdown, so it shows the name of the gear on each channel and nothing more. The **MIDI Monitor** key's **Channel** list is a *filter* on what arrives, and a name there tells

you whose traffic you are watching. In all three, renaming stays on the keys that actually address a channel, a controller or a note. One more thing worth knowing about **channel** names in particular: they show in every *list* — each panel's Channel row, the Monitor's filter, Panic's destination list — and never on a key face. An unnamed key names the message it sends (`CC127 ch16`), and that form is already at the ten-character limit the key face allows, so there is no room for a name as well; the number is the half you cannot do without.

- CC and program names are **scoped per channel** — CC 22 on channel 1 and channel 2 keep separate names and never mix. Note names are **global**, since a note number means the same thing on every channel.
- **Names in brackets are ours, not yours.** Where MIDI already has a name for a control number, the list shows it in brackets — `6 · (Data Entry MSB)`, `120 · (All Sound Off)` — so you can tell at a glance which values you have actually named. Type your own name for one and the brackets go: `6 · Gain`. Every list follows this, the read-only ones included, so the same number reads the same way wherever you meet it. Note names have no brackets, because `60` really is `C4` on any gear, while a control-number name only describes what the protocol does with it.
- **Note names** start from the musical name (`60 · C4`) and you can replace it — `36 · Kick` is the usual reason. Which octave numbering those built-in names use is a single global choice under **Naming** on any Note key: `Middle C = C4` for Roland gear and most DAWs, `C3` for Yamaha, or `C5`, which suits a few older sequencers. It changes the label only; the note number sent never changes.
- **Name map** — import a whole set of names at once, rather than typing them one at a time. Any key that offers naming has this row; the names are shared, so it does the same thing wherever you open it.MIDI Map Studio can build one for you from a Neural DSP mapping file: under **Names for MIDIRover**, press **Copy name map**. A map is also plain JSON, so anyone can write one by hand or share theirs. You can paste it in **either of two notations**. A name-map **JSON** document is what Map Studio copies and what one machine hands another. For typing by hand there is a plainer form, which needs no brackets, quotes or commas:

```
# my rig
[channel 1]
name = Amp
cc 22 = Drive
program 0 = Clean

[notes]
36 = Kick
```

Sections are `[channel N]` and `[notes]`; a line is `what = name`; `#` starts a comment and blank lines are ignored. Case does not matter, `:` works as well as `=`, and `pc` or `prog` mean `program`. If a line cannot be read, the import is refused whole and **names the line** — which is the reason this form exists, because a JSON error can only point at a character. You do not need this page to hand: the panel shows an example of **both** notations — press the small **?** beside the buttons — each with a button that copies it, ready to edit. Press **Import** — it **merges**: a name the document does not mention is left exactly as it was, so importing a map for one device never costs you the names you typed for another. A document that is not a name map, or is from a newer version of MIDIRover, is refused whole and nothing changes — you are told which, rather than being left with half a map applied. What you pasted stays in the box so you can correct it. Two more documents are refused for the same reason, and both matter most if you are*generating* a map rather than typing one: a **blank name** (a name with no text in it would quietly delete the name that was there, so leave the entry out instead, or use Clear), and the**same number written twice** in one section — `7` and `07` are the same controller, and a document that names it twice is saying two things about one address. Leading zeros themselves are fine: `cc 07` and `cc 7` mean the same thing, in either notation. A third refusal joins those two: a name containing a character with **nothing to draw** — a control character, a text-direction override, a zero-width space. These are invisible, so a name carrying one cannot be read, retyped or told apart from the name beside it, and a tool that pads or converts text is the usual way one arrives. MIDIRover names the character it found and the line it is on, so you can fix the document at the source. Ordinary text is untouched: spaces, punctuation, accented letters and non-Latin scripts are all names — `Bank P3`, `Amp Ch. 2` and `リード` all import. The last refusal is **length**: a name can be up to **40 characters**. That is not a limit invented for imports — it is the longest name the panel already lets you type into a key's **Name**box, and the rename boxes stop at 24, so a longer one in a document is asking for something you could not have created by hand. A key shows roughly a dozen characters anyway. As with every other refusal, MIDIRover says which name and how long it was, and changes nothing — it will not quietly shorten a name, because that would rewrite a document you may be sharing with someone else.**Export** goes the other way: it reads out the names you have already typed, into the same box and selected, so `Ctrl` + `C` copies them. That is how a set of names you built by hand leaves this computer — to keep, to move to another machine, or to give to someone with the same gear. One thing an export does *not* carry is a **source** block saying where the names came from, and that is deliberate: your profile is whatever you have merged — perhaps a pack for one device, a pack for another, and a dozen names you typed — so no single origin would be true of the result. If you are putting together a pack to share, add that block to the exported document by hand.**Clear** removes them all. Because importing *merges*, names from two different rigs otherwise accumulate with no way to separate them — Clear is how you start again. It asks twice (the button becomes **Really clear?**, and waiting cancels), tells you how many

it removed, and touches nothing but names: your MIDI port, your remembered channel, your colours and your Middle C setting all stay. **Export first if you might want them back** — that copy is the only undo. One thing to expect: keys on *other* pages keep showing their old names until you next open those pages. MIDIRover can only redraw the keys the deck is currently showing, and a key redraws itself correctly the moment it appears. An import also **tells you if the map renames one of MIDI's own control numbers** — CC 120*All Sound Off*, CC 0 *Bank Select* and a handful of others. It still imports, because plenty of gear really does drive its own parameters from those numbers; it is worth knowing because on most gear they do something structural rather than setting a value.

---

7 Neural DSP quick-start

1

Route MIDI in

Open your Neural DSP plugin (or its standalone) and set its MIDI **input** to the port MIDIRover sends on — your loopMIDI port on Windows, **MIDIRover** on macOS.

2

MIDI-Learn a control

Right-click a stomp or knob in Neural DSP and choose **MIDI Learn**. Press your MIDIRover **Control Change** key — it binds instantly.

3

Switch presets

In Neural DSP's MIDI Mapping window, add a **Program Change → Preset** entry per preset, then use Program Change or Bank Up / Down keys to jump or scroll tones. In a DAW the host has to forward Program Change (Ableton Live doesn't; Logic often doesn't) — the standalone app is the reliable path, or map presets to CCs there instead.

Neural DSP treats a learned CC on a button as a **toggle** — use **Fixed** mode for one value per press, or **Toggle** with **Feedback** on so the key light stays in sync. To flip a parameter between two settings without a dial, put **two** Control Changes on the **same CC** in **Fixed** mode with different values (e.g. a rhythm and a lead gain) — each jumps the parameter to its value. And CC **Toggle** keys sharing a **Latch group** name act like radio buttons, so the deck shows one active at a time. To see your whole MIDI map at a glance — or build a new one from scratch — open the MIDI Map Studio.

8 Troubleshooting

Nothing happens when I press a key

Open any MIDIRover key's settings and read the line under **MIDI Port** — it names the port and says whether anything is leaving the plugin at all. **Red**: nothing is being sent, and the line says why. **Amber**: MIDI is going somewhere other than the port this key is set to. **Grey**: MIDIRover is sending where you meant, so the problem is downstream — check that loopMIDI is running (Windows), that your software's MIDI *input* is the same port, and that the **channel** matches on both sides. Opening the settings is worth doing on its own: it makes MIDIRover re-check the port and reopen it.

MIDIRover has gone grey — the Stream Deck app has disabled it

This is not a licence problem, a broken install, or anything to do with your MIDI setup: it means MIDIRover **failed to start**, and the app has stopped trying. A plugin that dies while starting is relaunched automatically every few seconds, and after **twelve** failed launches the app gives up and greys it out — so this appears a couple of minutes after the first failure rather than the moment something goes wrong. It is nearly always a machine still waking up: MIDIRover waits and retries while the app starts, and the disable only lands when whatever it is waiting for takes longer than that.

The way back is to restart the Stream Deck app for real. The disabled mark lives in the running app and is written nowhere on disk, so a genuine restart always clears it — there is nothing to repair, nothing to reinstall, and no reason to delete your keys. What does *not* work is closing the window: the app keeps running in the Windows notification area (or the macOS menu bar), and MIDIRover stays grey however many times you bring the window back. Quit it properly — right-click the tray icon and choose **Quit** on Windows, or **Quit Stream Deck** from the menu bar on macOS — then start it again. If the actions are still grey after that, the app never really quit; restarting the computer is the sure way and costs less than it sounds, because your keys and their settings are stored and come back exactly as they were.

If it keeps happening, the file to send is `startup-error.log` — MIDIRover writes it beside its own bundle, in the `bin` folder of its plugin folder (`%appdata%\Elgato\StreamDeck\Plugins\com.midirover.plugin.sdPlugin\bin` on Windows, or `~/Library/Application Support/com.elgato.StreamDeck/Plugins/com.midirover.plugin.sdPlugin/bin` on macOS). That file is the only artifact this failure leaves: the ordinary MIDIRover log is *empty*, because the plugin never got far enough to open it. Send `startup-error.log` and say what the machine was doing at the time — waking from sleep, first login, a fresh install — and that is enough to work from.

Where the log file is, and what to send if you report a problem

MIDIRover writes a log beside the plugin, and on Windows the real file lives at `%LOCALAPPDATA%\midirover-logs` — paste that into the address bar of any Explorer window. (The `logs` folder inside the plugin itself is a shortcut to the same place; it is arranged that way so the log is never inside a synced folder, where it would be locked or duplicated.) On macOS the log sits in the plugin's own `logs` folder under the Stream Deck app's Plugins directory. Take the **newest** file: the log rotates every time the plugin restarts, so the one you want is usually `.0.log`, and timestamps inside it are UTC rather than your local clock. If you are reporting something, the last hundred lines around the moment it went wrong are far more useful than the whole file — and a **MIDI Monitor** key on screen at the time answers "did anything arrive at all", which the log alone often cannot.

The plugin isn't receiving in Neural DSP

Confirm the **channel** matches on both sides, and that you MIDI-Learned the right control. Watch incoming MIDI with a monitor if unsure.

I can't find the MIDI Dial action

The dial actions need a model with dials; the **Stream Deck +** and **+ XL** are the two Elgato documents with a touch strip to display the value on. On other models the dial action is hidden by design — use the key actions (e.g. CC **Sweep** mode to ride by hand).

The port is there, opens fine, and nothing crosses it

A loopMIDI port created — or re-created — *while other apps already held loopMIDI ports open* can come up listed but **non-functional**. Every screen looks correct and no error appears anywhere; the only tell is a MIDI monitor showing **zero** traffic. **Restart loopMIDI** (quit it fully and reopen) — you don't need to restart the Stream Deck app, because MIDIRover notices the port coming back and reopens by itself. Re-adding the port while apps are attached often reproduces it. Seen twice on one Windows machine. Detail in the troubleshooting guide.

I deleted and re-created a virtual port — do I need to restart?

No. If the port disappears and comes back, MIDIRover notices within about three seconds and reopens *both* its MIDI output and its Feedback input by itself, with nothing to touch. (That is for a port you picked *by name*. If you left the port on **Auto**, MIDIRover picks again on your next key press rather than in the background.)

My port vanished from loopMIDI — why does the key still look fine?

It shouldn't any more. Open any key's settings and the line under **MIDI Port** turns red and names the port that went missing, and the dropdown shows it marked **(not present)**. MIDIRover deliberately does *not* switch you to some other device when your saved port disappears — quietly rerouting a rig into whatever else happens to be plugged in is worse than sending nothing — so nothing is sent until the port returns (it reopens by itself) or you pick another one. Re-create it in loopMIDI with the **same name** and it reconnects on its own. If the port is gone for good — renamed, or a device you no longer have — choose **Auto** at the top of the dropdown: that releases the saved name and lets MIDIRover pick again.

Windows: no ports listed

Open loopMIDI and create at least one port, then reopen the key settings so the list refreshes.

None of the above — it still doesn't work

That is worth telling me about: an unexplained failure here is a bug in MIDIRover or a gap in this manual, and both are mine to fix. Report it with what you pressed, what you expected, and what the line under **MIDI Port** says — those three sentences are usually enough to identify it.

The key shows a fixed picture and never updates

If you set **your own image** on a key in the Stream Deck app, that image *wins* — Stream Deck stops letting the plugin draw, so the live graphics freeze. Nothing is broken and no error appears; the key just never changes again. It affects everything MIDIRover draws: the Sweep slider and knob, the Toggle on/off graphic, the Bank and Program numbers, and the MIDI Readout value. **Clear the custom image on that key** in the Stream Deck app (reset its icon back to the action's default) and the live graphic returns immediately. To keep a personal look, use MIDIRover's own **colour** settings instead — a per-key palette or the deck theme — which restyle the graphic rather than replacing it.

9 Support & feedback

Something not working, or a key you wish existed? Report a bug or ask for help — it goes straight to the developer, and a clear report is the fastest way to get a fix.

MIDIRover is free and made by an independent developer who plays guitar. If it saves you setup time,buy me a coffee.