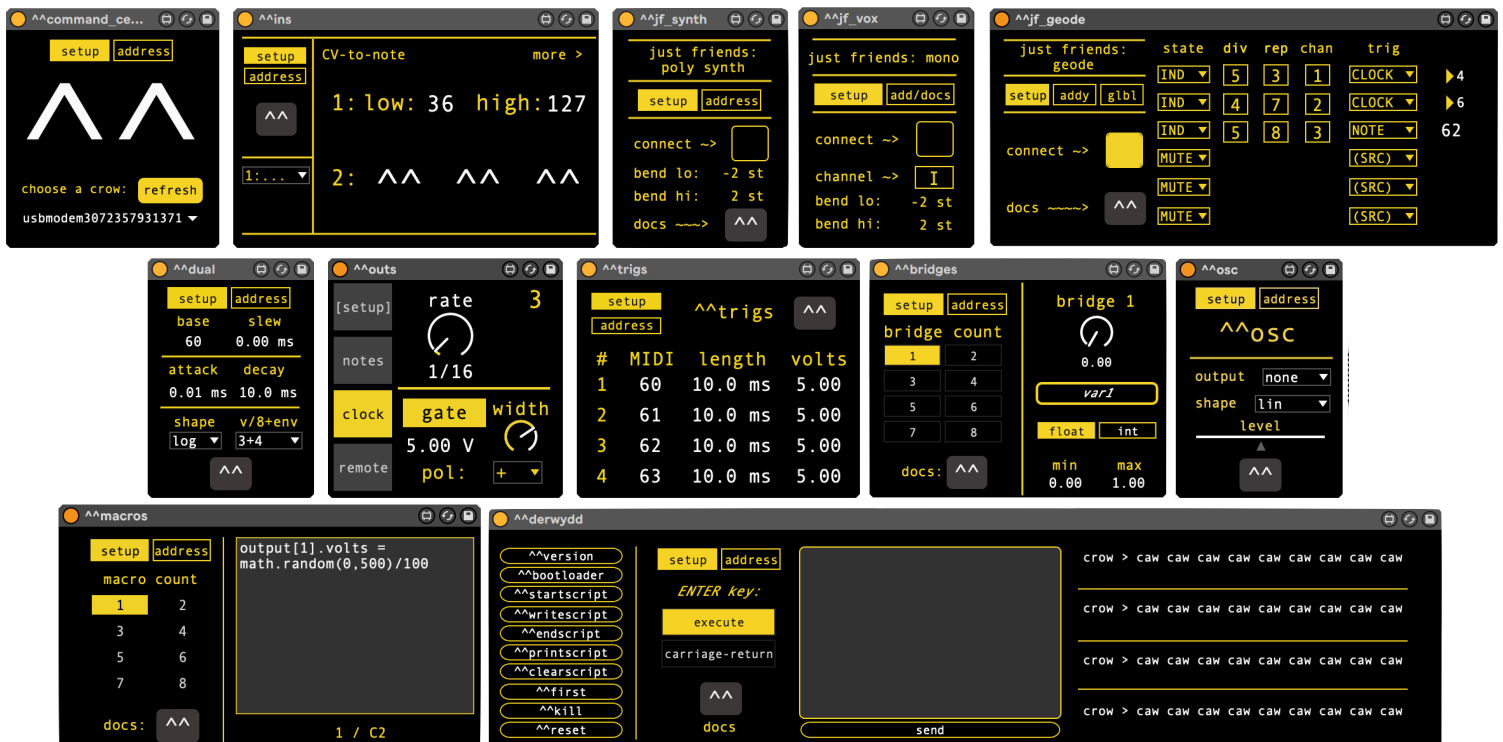




## m4l devices for crow



crow 4.0+  
updated: 230310

## `crow m4l: getting started`

---

`crow` is a 2hp eurorack module made by whimsical raps + monome which connects to noins and computers running Max, Max for Live, and other serial-enabled applications through USB.

This guide covers a bundle of Max for Live devices made by monome + whimsical raps to integrate Ableton Live and your modular synth through a variety of use-cases.

### requirements:

- `crow` module with firmware v4.0+
- Ableton Live Suite (tested with 11, though other versions may work) (Mac/Win)
- Max for Live

To get started, visit <https://github.com/monome/crow-max-and-m4l> and select "Clone or download". This will download everything you need to get started with `crow`, Max, and Max for Live. After downloading the entire `crow-max-and-m4l` repo, extract the zip file and you should get two unique folders: `crow_max` and `crow_m4l`.

Place the `crow_m4l` folder wherever you'd prefer it living longterm on your hard drive. Then, open Live and drag the folder into Live's browser, under PLACES. If you are updating a previous installation, just replace the previous `crow_m4l` folder's contents with the new files.

Need help? Want to share what you're making? Visit the [crow m4l thread](#) on lines.

---

**^^command\_center** routes messages between Live and `crow`. The devices will not connect to `crow` unless **^^command\_center** is properly initialized.

### voice control:

- ^^dual**: translate MIDI data from Live to v/8 and variable envelope voltages
- ^^ins**: translate incoming v/8 and triggers through `crow` to MIDI notes
- ^^osc**: translate MIDI data from Live to audio-rate oscillators through `crow`
- ^^jf\_synth + ^^jf\_vox**: MIDI-to-i2c output to play a connected Just Friends module

### modulation + events:

- ^^outs**: a single MIDI-to-CV output device that collects multiple utilities
- ^^ins**: translate incoming CV through `crow` to useful MIDI data for Live
- ^^trigs**: a four-channel MIDI trigger-to-pulse device, useful for rhythmic events
- ^^tport**: sync Live to your modular system with transport + tempo controls

### crow programming:

- ^^bridges**: translate multiple mappable knobs in Live into data for `crow`
- ^^derwydd**: send Lua code to `crow` to execute + modify `crow` code in real-time
- ^^macros**: store code snippets which can be sent to `crow` as macros on the fly

^^command\_center, when properly initialized, connects the other Max for Live devices connect to crow.

^^command\_center setup:

- load onto any MIDI track
- select your connected crow device from the dropdown
- don't see your crow? hit [refresh]

## CONNECTING MANY CROW? EACH NEEDS AN ADDRESS!

*nb. If you are NOT connecting more than one crow to Live, you do not need to perform the actions outlined in this section. They do not apply to a single-crow configuration.*

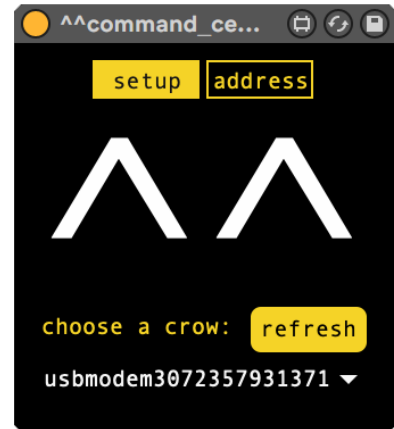
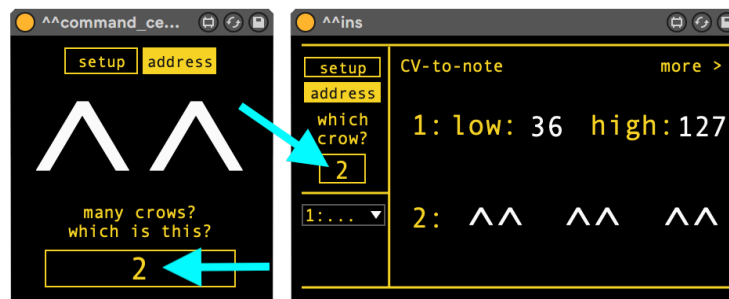
If you'd like to connect more than one physical crow module to the m4l devices, you'll need to instantiate a unique ^^command\_center for each crow and give it a Live address. You'll also have to tell the other m4l devices to which address they should send messages. Don't worry, these addresses will all save with your Live Set.

Every ^^command\_center defaults to address 1. To make sure messages from the m4l devices get to the right crow, toggle from [setup] to [address] and you'll be able to change ^^command\_center's address.

A typical two-crow setup:

- connect each crow to your computer through USB
- instantiate two ^^command\_center devices
- use the dropdown on the [setup] page to connect each ^^command\_center to a different crow
- toggle to [address] and make sure that one ^^command\_center has address 1 and the other has address 2
- on each of the other m4l devices, toggle [address] to direct the flow of traffic from the device back to the correct ^^command\_center.

*If I want to use ^^ins with crow 2, I would have to specify 2 as ^^ins address:*



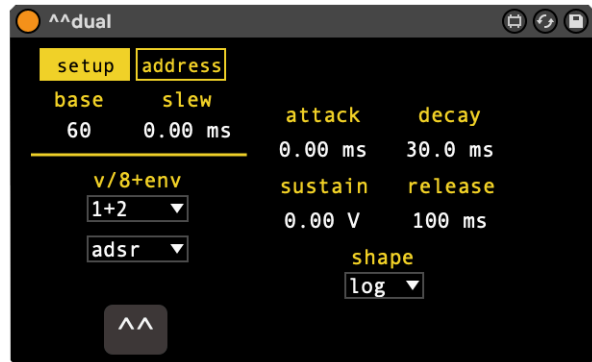
voice control: ^^dual + ^^ins

---

**^^dual** translates MIDI note data from Live to v/8 and envelope voltages. Load it onto any MIDI track and arm it for recording or set the track's monitoring to *in*.

#### overview

base        the MIDI pitch which equals 0V  
slew        add glide between MIDI pitches  
attack      define output envelope's attack time  
decay       define output envelope's decay time  
v/8 + env   identify which duo of outputs to use for v/8 and envelope  
shape       specify envelope shape: log, lin, or exp



#### crow outputs

output 1 or 3    v/8 from MIDI pitch  
output 2 or 4    envelope triggered from MIDI note-on

---

**^^ins** translates incoming CV to MIDI data. There are two modes – Mode 1 can be used to sequence a synth or VST in Live.

#### MODE 1: CV-to-note

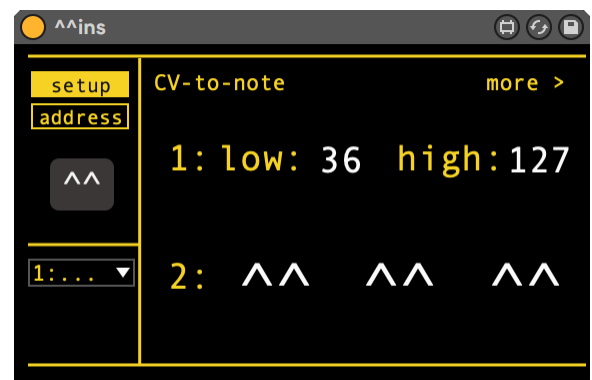
CV-to-note mode translates incoming v/8 and triggers to create MIDI note events.

#### overview

low        set the desired floor for CV-to-MIDI pitch, default is 0V == 36  
high       set the desired ceiling for CV-to-MIDI pitch, default is 127  
^^        trigger indicators (passive)

#### crow inputs

input 1        expects v/8  
input 2        expects trigger (5V)



voice control: ^^osc

---

**^^osc** translates MIDI note data from Live to audio-rate oscillators on crow. Load it onto any MIDI track and arm it for recording or set the track's monitoring to *in*.

#### overview

output    the output you'd like to oscillate  
shape    the base waveshape (timbre)  
level    controls output amplitude, but also  
         provides a nice overdriven effect  
         to the chosen shape



*nb. while it is encouraged to modulate the 'level' parameter of a single ^^osc voice (eg. with an Envelope Follower), this can create a lot of messages to crow which may produce instability with more than 2 instances of the device*

voice control: ^^jf\_synth + ^^jf\_vox

---

**^^jf\_synth** and **^^jf\_vox** are two devices which send MIDI data from Live through crow to control an i2c-connected Just Friends module. Load it onto any MIDI track and either arm it for recording or set the track's monitoring to *in*.

#### More info about i2c

#### More info about Just Friends

**^^jf\_synth** addresses Just Friends as a 6-voice polyphonic synth. Notes are distributed I -> 6N, depending on how many are held down. Great for traditional keyboard playing + chords. Just Friends can be in *transient* or *sustain* mode.

#### overview

- connect toggle to connect to Just Friends  
*nb. if you see a ✓, Just Friends is already connected!*
- bend lo lowest pitch bend message target
- bend hi highest pitch bend message target



---

**^^jf\_vox** addresses individual voices of Just Friends. It is great for monophonic sequencing. It is particularly rewarding to address many individual voices at once. Just Friends can be in *transient* or *sustain* mode.

#### overview

- connect toggle to connect to Just Friends  
*nb. if you see a ✓, Just Friends is already connected!*
- channel the channel (I -> 6 or all) which you'd like to play  
*nb. engage on only **one** copy of ^^jf\_vox, otherwise Just Friends will disconnect*
- bend lo lowest pitch bend message target
- bend hi highest pitch bend message target



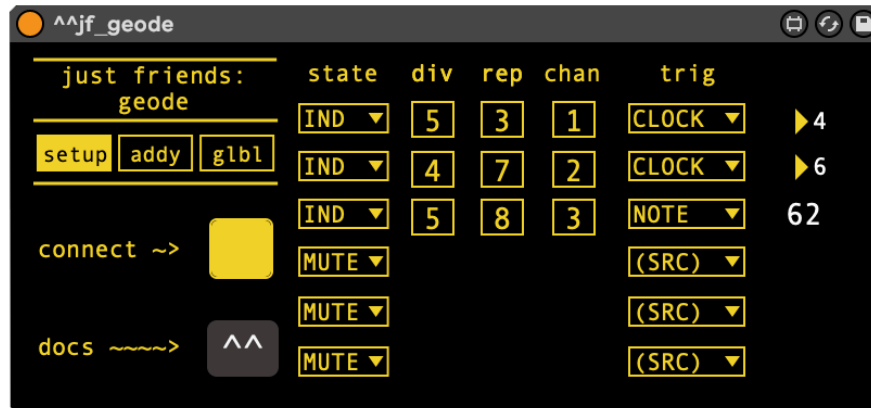
rhythm control: ^^jf\_geode

---

**^^jf\_geode** controls GEODE mode on an i2c-connected Just Friends module. Load it onto any MIDI track and either arm it for recording or set the track's monitoring to *in*.

### More info about GEODE

**^^jf\_geode** addresses Just Friends as a 6-voice tempo-responsive polyphonic rhythm machine. There are six sequencer channels, which can be sent to a specific hardware channel (IDENTITY -> 6N) or allocated in round robin. Just Friends can be in *transient*, *sustain*, or *cycle*.



### **overview**

#### setup

connect    toggle to connect to Just Friends  
          *nb. if you see a ✓, Just Friends is already connected!*

state      **IND**ividual or **RoundR**obin

div        how many ticks per measure (4 beats), eg. 4 means 1 tick per beat

rep        number of times to retrieve the envelope (-1 to run forever)

chan       select the channel to assign the rhythmic stream (0 for all)

trig       trigger the rhythmic stream either by a clock beat or a MIDI note

addy      for multi-crow setups, specify which crow to communicate with

#### glbl

q.div      quantize Geode events to *divisions* of a measure – if you need your rhythms to stay on a regular grid, activate that grid with q.div

clk rate   specifies the division of the main clock which CLOCK trig = 1 adheres to

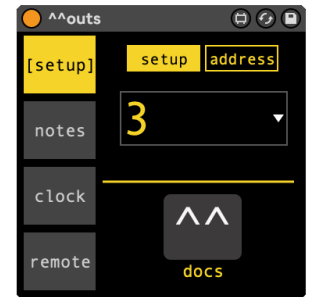
run?       enable RUN mode control

volts      applying RUN voltage emphasizes every 2nd then 3rd then 4th event, however all the in-between beats are available too. subtle shifts allow for a variation of groove with nothing but volume manipulation.

**^^outs** is a Swiss Army device – it holds a number of useful output utilities. Load it onto any MIDI track and arm it for recording or set the track's monitoring to *in*.  
*nb. you can instantiate this device up to 4x in a Live set, for each crow output*

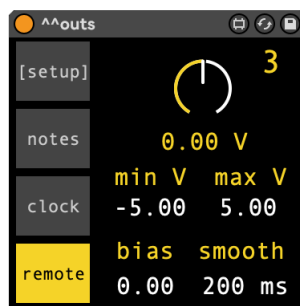
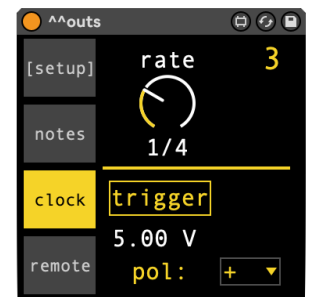
### overview

(out) identify which hardware output you'd like to use  
*nb. the device will display the selected output in the top right corner of the module screens*



**notes** *decoupled pitch cv or note-on trigger*  
[dropdown] choose v/8 or trigger signal  
base the base point for MIDI-to-CV conversion, default is MIDI note 60 = 0V  
slew adds glide between notes, default is none  
length when in trig mode, length of the trigger pulse

**clock** *clock-synced pulses at variable rates*  
rate clock pulse rate, synced to Live's transport  
trigger set the max voltage for the signal, default 5V  
gate toggle behavior for trigger which will reveal % duty cycle of current clock rate, default 50%  
polarity whether triggers are a burst of voltage (default) or an absence of voltage in a continuous on-state



**remote** *a knob which sends any movement out as CV*  
min V + the min/max voltage the knob can put out when the needle is far-left, default min: -5V, max: +5V  
max V  
offset adds voltage to the knob's current position, default 0V  
smooth adds glide between knob values, default 50ms

*nb. to send an LFO to any of crow's outputs, map the **remote** knob to Live's built-in LFO device (or any of the community devices you prefer)*



modulation + events: ^^trigs + ^^ins

---

**^^trigs** converts note-on events for up to four different MIDI notes to voltage triggers from crow. Helpful for converting drum-centric sequences to pulse events.

#### overview

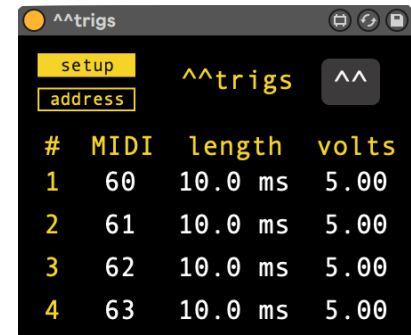
MIDI specify which MIDI notes should have note-ons translated to triggers

length the length of the trigger pulse

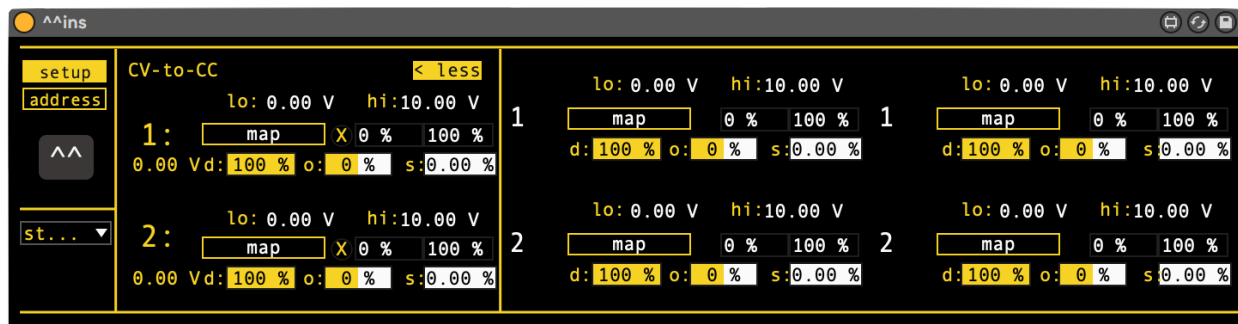
volts the voltage peak of the trigger

*nb. if you don't want to use this device for all four outputs, just set the unused outputs' MIDI to 0 to avoid accidentally triggering the device*

---



**^^ins** MODE 2: streams, CV-to-CC



#### overview

low set the desired floor for voltage-to-CC, default is 0V

high set the desired ceiling for voltage-to-CC, default is 10V

map map the incoming voltage to any MIDI-controllable parameter in Live

d depth of cv-to-cc

o offset the scaling of received cv values

s slew the cv-to-cc conversion, to soften clickiness/steppiness when mapping to audio parameters (like panning, filter cutoff, gain, etc)

> more open an additional 6 channels of mapping

#### crow inputs

input 1 expects lfo / continuous voltage source

input 2 expects lfo / continuous voltage source

crow programming: ^^bridges + ^^derwydd

---

**^^bridges** translates multiple mappable knobs into data for a crow script; the primary use case is remote control over variables in a currently running script.

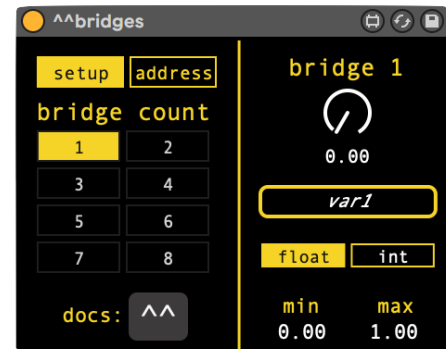
#### overview

bridge count    adds/removes bridges

varX            type in a name matching a  
                 variable in your script

knob            a remote control for varX

min / max       range of knob's affect on varX



tips:

- try automating the knob
- try MIDI mapping the knob with a MIDI controller
- try mapping the knob with another Max For Live Device

---

**^^derwydd** gives you access to crow's Lua read-eval-print loop, much like druid. Send Lua code to crow to execute on the fly, allowing you to modify crow's behavior in your real time. You can also use it to upload new scripts, erase scripts, and more.



#### overview

^^ commands    executes system crow commands, hover over to learn more

ENTER key       toggle ENTER key behavior to execute codebox or add new line to  
                 allowing you to send more complex multi-line code snippets to crow

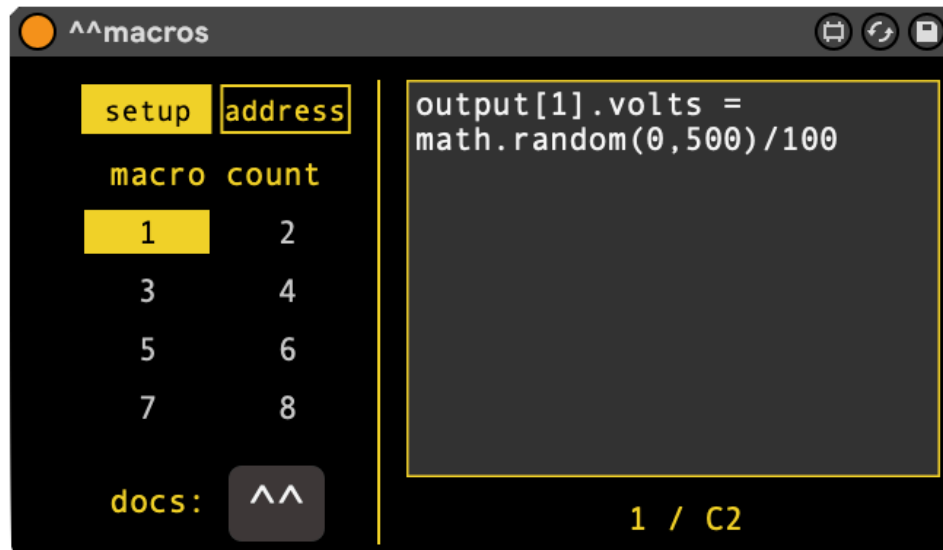
codebox        type in Lua code to send to crow

send            execute code in codebox

tips:

- use the up and down arrow keys to scroll through recently entered code snippets
- printouts and error messages from crow are displayed to the right of the codebox

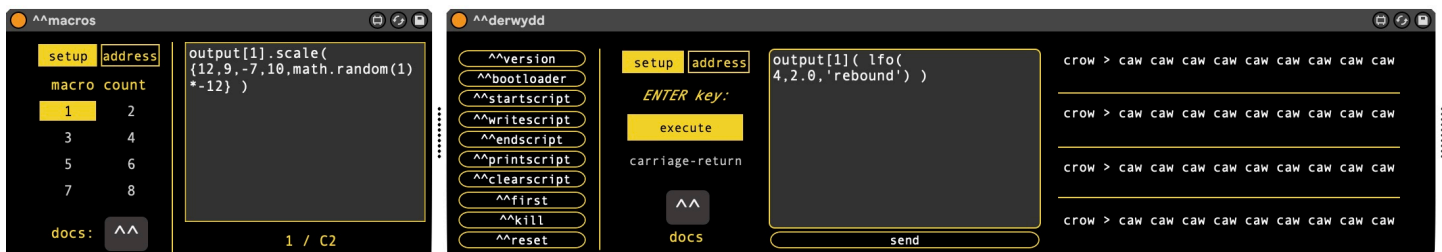
**^^macros** sequences code snippets. Each snippet can be sent by selecting the device and pressing the corresponding number key on your keypad, or by sending the device the corresponding MIDI note. Code is saved with your Live set.



#### overview

- macro count            adds/removes macros
- codebox                type in Lua code to send to crow
- # / MIDI note         send the corresponding MIDI note to the device, or press the corresponding key on your keyboard to execute codebox  
*nb: when using number keys, make sure the device is highlighted*

The best practice with **^^macros** is to use one macro per line of code you wish to execute on-demand. For one-off commands, we recommend using **^^derwydd**, eg:



`output[1].scale( {0,3,5,9,2,10} )` – send via **^^derwydd**  
`output[1].volts = math.random() * 3` – send via **^^macros**

`output[1]( { to(1,1), to(1.8,2), to(2.4,3),to(0,0.7) } )` – send via **^^derwydd**  
`output[1].scale( {4,11,2,0,7,6} )` – change values on-the-fly and send via **^^macros**

`output[1]( lfo( 4,2.0,'rebound') )` – send via **^^derwydd**  
`output[1].scale( {12,9,-7,10,-12} )` – change values on-the-fly and send via **^^macros**