

## MAX/MSP By Example

by Karen Collins [www.gamesound.com](http://www.gamesound.com)

### LESSON #6 BUILDING UP A RHYTHM: The **Metro** object and **counter**

Let's work on building up a basic rhythm using a **metro** object. A **metro** works like a metronome—it sends an output at a regular pace.

For instance, in Figure 6.1, when the **toggle** is clicked, **makenote** will receive a regular message to make a note 24 at velocity 100 and duration 100.

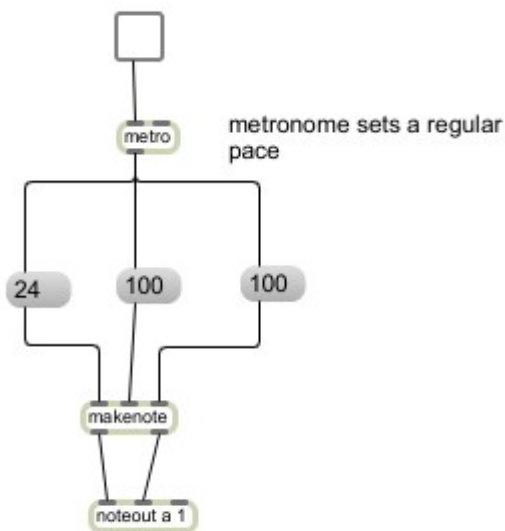


Figure 6.1 Creating a very basic beat

Musicians commonly want to work in beats per minute (BPM) rather than milliseconds (ms). However, the **metro** object in MAX takes ms rather than BPM as its default. We can turn our BPM into ms using this handy mathematical equation. Since BPM is number of beats per 60 seconds, and a millisecond is 1000 parts of a second, we want to divide 60 000 (60 \* 1000) by the input BPM. Note that in Figure 6.2 we're using a **float** rather than an **integer** to get exact timing.

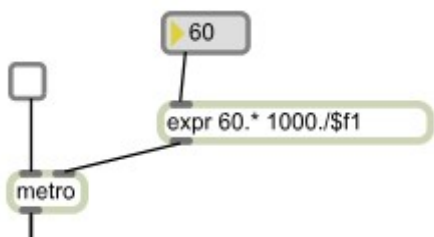


Figure 6.2 Turning a variable (**number box**) BPM into milliseconds.

We can now take our earlier patch and have an adjustable BPM to our bass note (Figure 6.3).

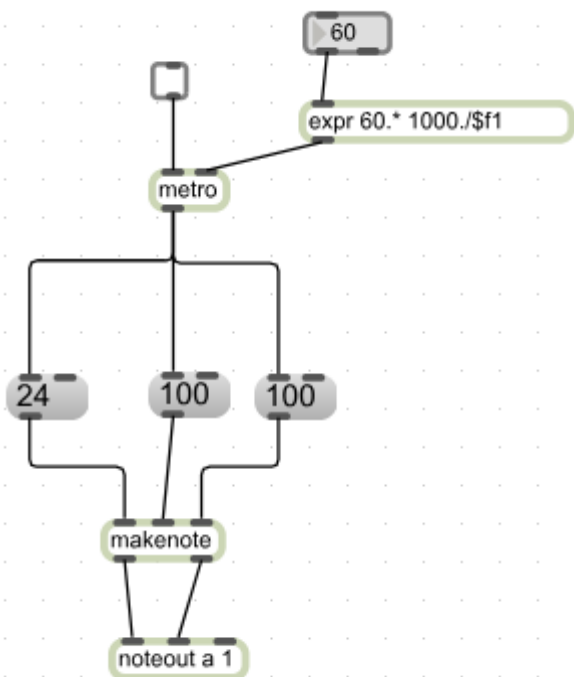


Figure 6.3 BPM added to our earlier patch.

But a simple BPM beat is pretty boring. Before we get all creative, let's add a "snare" at half the BPM rate. We'll stick with MIDI notes for now, even though we probably wouldn't use them if we were making a finished product, since they don't make great drums (Figure 6.4).

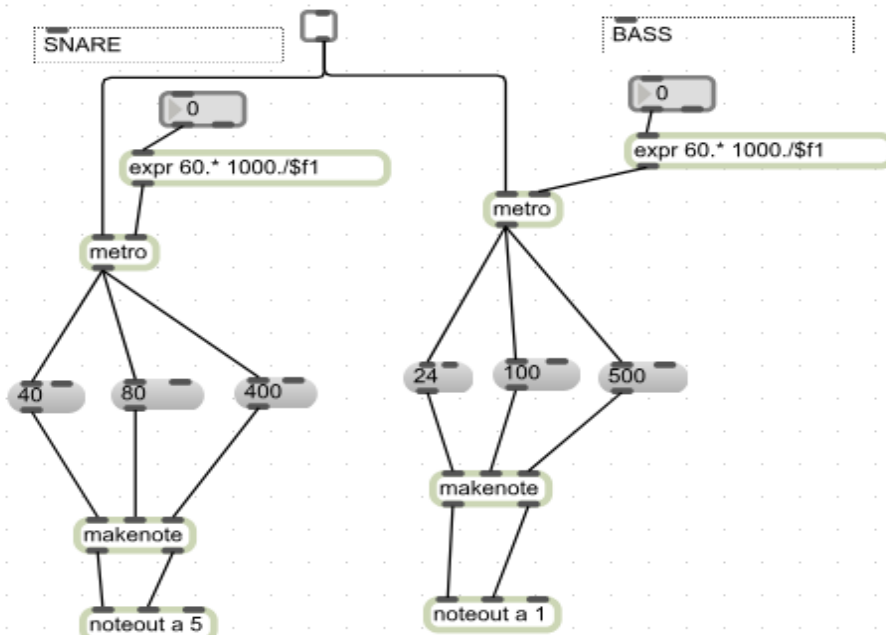


Figure 6.4 Our basic rhythm patch with "bass" and "snare"

OK, so now we have a very basic rhythm patch. It's quite boring, though, so let's learn some new tricks! First, let's use a **counter**. We can count how many times the metronome has clocked by including a **counter** with a number output. Figure 6.5 counts how many seconds have gone by (the number of times **metro** has sent out a bang).

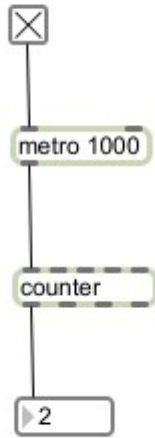


Figure 6.5 A basic **counter**.

Now hold onto that thought. What if we want something to happen when the **counter** reaches a certain number-- that is, in this instance, we want it to play a different rhythm when the count (beat) gets to 13? The first thing we want to do is check the output and see if it is greater than 12, then have that send out a trigger. The **trigger** functions like a binary number-- 1 is on, 0 is off. By turning it "on" when it hits 13, we can have the **trigger** on (= 1, or true) (Figure 6.6)

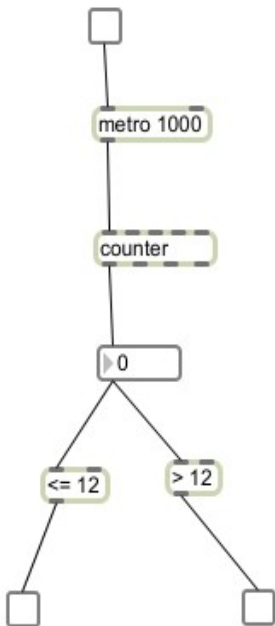


Figure 6.6 A **counter** sends the signal to a new direction on the 13<sup>th</sup> beat.