



John Barr, Back Page Editor
Department of Computer Science
Ithaca College
953 Danby Road
Ithaca, New York 14850 USA
barr@ithaca.edu

Converting MIDI Notes to ABC Notes in Pencil Code

By Jesse M. Heines

Comparing Scratch [4] to Pencil Code [3] for teaching computing through music immediately reveals a major difference: Scratch represents notes as MIDI numbers, while Pencil Code represents notes as letters using ABC notation [1]. To someone who reads music, Pencil Code is clearly preferable because it is far easier to map, for example, a major third to C and E than 60 and 64. But to those who generate music algorithmically, ABC is not as easy as MIDI. For example, a major third in the key of D is not D and F, it is D and F#. Using MIDI values, however, the interval is always 4 semitones: 62 (D) + 4 = 66 (F#), just as before.

The Problem: Pencil Code is written in CoffeeScript [4], “a little language

that compiles into JavaScript.” Thus, one would think to use the `String.fromCharCode` function to convert numeric MIDI to alphabetic ABC. Unfortunately, the conversion is not that straightforward, because ABC notation is based on a C scale, with A and B above G. That throws out a simple conversion. The 4th octave, the one that begins on middle C, is represented by all capital letters: C D E F G A B. For the 5th octave, apostrophes are added: C' D' E' F' G' A' B'. For the 6th octave, two apostrophes are added, and so on to the higher octaves. For the 3rd octave, the one below middle C, commas are added: C, D, E, F, G, A, B. And, as you can probably guess from the pattern, for the 2nd octave, two commas are added and so on to the lower octaves.

A Solution: If one wants both the clarity of ABC notation and the algorithmic power of MIDI values, a conversion is indeed necessary. The code below runs in Pencil Code and plays the chromatic scale from C3 to C6 by passing MIDI values 48 (C3) through 84 (C6) to a `MIDItoABC` function that is generalized to convert any MIDI value to its proper representation in ABC notation and play it for a specified duration.

A Note on ABC Duration Notation: Unrelated to MIDI but germane to ABC, duration strings are based on quarter notes, which have a duration of 1. A half note has a duration of 2, and a whole note a duration of 4. Going the other way, an eighth note has a duration of 1/2, designated as “/2”. Similarly, a 16th note has a duration of “/4”, and a 32nd note has a duration of “/8”.

The code below includes a test routine to play the chromatic scale from C3 to C6 with random note durations from a half note (“2”) to a 32nd note (“/8”). ❖

```
1 # PencilCode script to convert MIDI numbers to ABC notation notes
2 # by Jesse M. Heines, UMass Lowell, Version 4.2, February 2, 2016
3
4 # define an array of chromatic notes beginning with middle C
5 MIDImap = ["C", "^C", "D", "^D", "E", "F", "^F", "G", "^G", "A", "^A", "B"]
6
7 # function to convert a MIDI number to an ABC note
8 MIDItoABC = ( midi, duration=1 ) ->
9   # test that a MIDI number has been supplied
10   if midi? and parseInt(midi) != NaN
11     # compute desired octave (octave 4 starts at middle C)
12     octave = parseInt((midi-12)/12.10)
13     # initialize the suffix string of commas or apostrophes
14     suffix = ""
15     # add , or ^ as appropriate for desired octave
16     if octave < 4
17       suffix += "," for [1..abs(octave-4)]
18     else if octave > 4
19       suffix += "^" for [1..octave-4]
20     # return value
21     MIDImap[midi % 12] + suffix + duration
22
23 # test: play chromatic C3 to C6 with random note durations
24 for k in [48..84]
25   duration = random ["2", "1", "/2", "/4", "/8"]
26   play MIDItoABC(k, duration)
```

References

1. ABC Notation; <http://abcnotation.com>. Accessed 22 February 2016.
2. CoffeeScript; <http://coffeescript.org>. Accessed 22 February 2016.
3. PencilCode; <http://pencilcode.net>. Accessed 22 February 2016.
4. Scratch; <http://scratch.mit.edu>. Accessed 22 February 2016.

Jesse M. Heines
UMass Lowell—Computer Science
One University Avenue
Lowell, Massachusetts 01854 USA
heines@cs.uml.edu

DOI: 10.1145/2927021

Copyright held by author.