**MML to MIDI Language Specification**

**Introduction**

The music macro language (MML) is a music description language that has been in use since 1978, although this was an early version. There has never been an official specification, so each implementation varies slightly, and over the years the language has evolved. The MML to MIDI converter uses a version of the MML derived largly from “Classical MML” with some “Modern MML” features present. Some new specific commands are included also, and some commands are changed where necessary. This document outlines this version of the MML used by the program, and contrasts it to existing syntax where it significantly differs.

All commands are terminated by a new line (either \n or \n\r).

**Comments**

Comments are started with two hash characters, “##”. This makes the remainder of the line a comment; any more hashes found on the line have no effect. Two hashes are used because single hash starts a meta command.

**Playing Notes**

Playing notes is done by starting a line with the word “play”, then entering the notes after a space. Any spaces aside from this initial one are ignored, and the command is ended with a new line. An example of the music command is:

play c5e5g5

**Note Syntax (update needed because of ‘l’)**

Notes are written as the note name followed by the length of the note as a digit – each value for this digit represents a musical note length, which can be seen in the table below. A rest is represented by the note name “r”. To play an accidental note a “+” or “-”, respectively, is added after the note name and before the note length. Accidentals applied to rests are ignored.

|  |  |  |
| --- | --- | --- |
| MML Note Value Number | Musical Note | |
| American Notation | Name |
| 0 | 1/32 | Demisemiquaver |
| 1 | 1/16 | Semiquaver |
| 2 | 1/16 + 1/32 | Dotted semiquaver |
| 3 | 1/8 | Quaver |
| 4 | 1/8 + 1/16 | Dotted quaver |
| 5 | 1/4 | Crochet |
| 6 | 1/4 + 1/8 | Dotted crochet |
| 7 | 1/2 | Minim |
| 8 | 1/2 + 1/4 | Dotted minium |
| 9 | 1 | Semibreve |

To alter how each note is played, there are some of commands entered with the notes. These are listed below (where square brackets and their contents are not literal):

* o[digit] Set the octave each following note is played in. The digit represents the scientific pitch notation (SPN) number of the desired octave. All notes entered before this command is entered are played in the 4th SPN octave (“A” will be 440 Hz.)
* < Shift the octave down by one.
* > Shift the octave up by one.
* v[digit] Set the volume of the following notes. By default, notes will play at 100% volume.
* p[number from 0 to 11] Transpose all the following notes up by the number following ‘p’ semitones. The default setting is 0.
* l[digit] Set the default length of the follow notes to the digit. The initial default length is 5. Note that this does not affect the ‘v’ or ‘o’ commands.

In modern MML there are also the commands “t” and “l”. “t”, which sets the tempo, is not included because MIDI only supports one setting of the tempo for an entire song. “l”, which sets the default length of a note (for when it is not specified), is not included because, although it makes the notation less verbose, it also makes it less obvious to read.

**Meta Commands**

These commands are entered on their own lines only once and are all preceded by a single hash. These tell the converter how the entire file should be played and add information to the MIDI file. They are listed below:

* #tempo [BPM] – set the tempo in BPM of the track (where a beat is a crochet.) This should be set the same in each MML track file when combining them into one MIDI file. If multiple tempo commands are found in the file, only the latest one will be used. The default tempo is 120 BPM.
* #instrument [general MIDI patch number] – set the instrument the file should be played with. If multiple instances of this command are found, only the latest will be used. The default instrument is a piano (GM patch number 0.) This command is not present in other MML versions because it is only useful if the file is being converted to a MIDI file.
* #name [name] – set the name of the track. This is put verbatim into the MIDI file in a track name meta event, and can be very useful when altering the MIDI file directly.

**Macros**

A macro in this version of MML is written as below (on it’s own line):

$c v9o4c5

The dollar sign shows that this is a macro definition, and the letter following this is the “name” of the macro. The text after the dollar sign and letter replaces any other instance of the macro name found. A limitation of this notation is that there are only 26 possible macro names, but it is done this way to be more compatible with other versions of MML. If a macro is defined more than once in a file an error will be raised.

**Full Example**

To conclude the document, a short example of a valid MML file is shown below.

|  |
| --- |
| ##Example comment  #name piano0  #instrument 0  #tempo 120  $c o4c5d5e5f5g5a5b5o5c5  play v9$c |