# RUN COMMAND:

To run your melody, please open a command line, change your current directory to where 'MusicGenerationClient' folder is (eg: E:\code\MusicGenerationClient), then type the commands in the console:

erlc user\_case.erl

erl -noshell -s main start -s init stop

Then press Enter to send the command to run.

To stop your melody, please type the command in a console:

erl -noshell -s main stop -s init stop

Then press Enter to send the command to run.

# FUNCTIONS:

This part is used to compile, users can ignore it.

-module(api).

-behaviour(melody).

-compile(export\_all).

## FUNCTION ONE: beats\_per\_minute()

Here users can change the bpm/tempo of the melody. Bpm value can be any number between 60 and 180.

Here are "typical" tempo ranges for a number of common genres:

Dub: 60-90 bpm

Hip-hop: 60-100 bpm

House: 115-130 bpm

Techno/trance: 120-140 bpm

Dubstep: 135-145 bpm

Drum and bass: 160-180 bpm

Example 1:

beats\_per\_minute() ->

60.

Example 2:

beats\_per\_minute() ->

100.

Example 3:

beats\_per\_minute() ->

128.

Below are Five track functions which allow you to define different kinds of tracks in your melody.

If you don't want to use any of them, simply comment or empty the **main body** of that function by adding "%" ahead of each line. NotePad++ also provides the operation of 'Block comment' the code block if you select the block then right click your mouse. REMEMBER: you should ONLY comment the code between the square brackets []! Or else, your music file will not be run properly.

Eg: comment the main body of unused function

sample\_pattern\_track() ->

[

% {"ambi\_glass\_hum", [x, o, x, o], echo, [{amplitude, 0.5}]},

% {"drum\_snare\_hard", [o, o, x], reverb, no\_envelope}

]

Or: Empty the main body of unused function

sample\_pattern\_track() ->

[]

## FUNCTION TWO: pitch\_based\_track()

Use this function to define some pitch-based tracks. Each tuple is one track. Each track consists of the following items:

**name, mode, duration, delay, instrument, sound, effect, envelope.**

Name: the name of this track. name needs to be surrounded by double quotes "". It also needs to be a unique one in the whole document.

Mode: the playing mode of this track. Users could choose from loop mode or non-loop mode. Use the keyword 'loop' to choose loop mode. Or a positive integer (like '2') to choose non-loop mode. The integer in no-loop mode will represent the iteration times of this track.

Duration: a float number with at most three decimal places which represents the duration of this track.

Eg, '3.25', '1.875', '2.0'.

REMEMBER: If you want to give an Integer, you must give it at least one decimal place to transfer this integer to float number.

Delay: how long you want this track to be delayed when playing. A float number with at most three decimal places should be given.

Eg, '2.89','4.0'.

REMEMBER: If you want to give an Integer, you must give it at least one decimal place to transfer this integer to float number.

Instrument: the instrument of this track. It needs to be surrounded by double quotes "". Please check the file ‘SoundsAndEffects.pptx’ to listen to the instrument samples.

Options:

beep, blade, dull\_bell, fm, growl, hoover, piano, pretty\_bell,prophet,saw,sine, square, supersaw

Sound: the notes and their beats in this melody. Each note and its beat are an independent tuple, Eg: {b1, '\_'}.

Note: the note could be a **combination of letter and number**, OR a **MIDI number** (a **float** number). If you want to have a rest note, just input, **rest**.

Combination of letter and number: The first letter represents the key in music, so it can be ‘a’, ‘b’, ‘c’, ‘d’, ‘e’, ‘f’ or ‘g’.

The second letter: ‘s’ means sharp accidental in music notes, while the letter ‘b’ means flat accidental in music notes.

The number after them means octave of this note, which ranges from 0 to 10.

Eg: rest, c0, cs0, db0, d0, ds0, eb0, e0, f0, fs0, gb0, as0, bb0, b0, c1, cs1, db1, g0, d1, gs0, ds1, ab0, eb1, a0, e1, f1, fs1, gb1, as1, bb1, b1...c10, ...b10,

60.24, 64.0, 65.

Beats: the duration of a note.

'\_' 0.25 beat

'\_\_' 0.5 beat

'\_\_\_' 1.0 beat

'\_\_\_\_' 2.0 beats

'\_\_\_\_\_' 4.0 beats

Effect: effect of this track. Please check the file ‘SoundsAndEffects.pptx’ to listen to the effect samples.

Options:

echo, ping\_pong, wobble, reverb, no\_effect.

If you don't want to have any effect, you could use no\_effect.

Envelope: envelope of the beats in this track. You can use it to modify the ADSR amplitude envelope of the beats in this track to control over their durations and volumes. See the detailed definition of ADSR in https://sonic-pi.net/tutorial.html#section-2-4.

It contains members of:

attack, decay, sustain, release, amplitude, attack\_level, decay\_level and sustain\_level

You could choose to define any of them. Each member's value is a float number, which can vary from 0.0 to 1.0.

Eg:

[{amplitude, 0.5}].

[{attack, 1.0}, {decay, 1.0}, {sustain, 1.0},{release, 1.0}, {amplitude, 0.1}].

[{release, 1.0},{amplitude, 0.0}].

[{attack\_level, 0.4}, {amplitude, 0.6}].

If you don't want to define any envelope attributes, you can input no\_envelope.

Example 1:

pitch\_based\_track() ->

[

{

"melody1",

loop,

3.25,

4.00,

"sine",

[

{60.24, '\_\_\_'},

{rest, '\_'},

{60.5, '\_\_\_\_\_'},

{rest, '\_\_\_'}

],

wobble,

[ {amplitude, 0.5}]

}

].

Example 2:

pitch\_based\_track() ->

[

{

"melody2",

2,

3.00,

7.67,

"piano",

[

{c4, '\_\_\_'},

{rest, '\_'},

{d4, '\_\_\_'},

{rest, '\_'}

],

echo,

[ {amplitude, 0.5}]

},

{

"melody3",

3,

3.00,

8.00,

"beep",

[

{64, '\_\_\_'},

{rest, '\_'},

{d4, '\_\_\_'},

{rest, '\_'}

],

no\_effect,

[ {amplitude, 0.5}]

}

].

## FUNCTION THREE: chord\_play()

Use this function to define some chord-based tracks. In each track, users could define a chord, play all the notes in this chord together.

Each tuple is one track. Each track consists of the following items:

name, mode, delay, instrument, note, chord\_type, beats list, effect, envelope.

All of the syntax of the common items follows the ones in pitch\_based\_track(). Please refer to the above.

Note: the tonic note for this chord.

Chord\_type: the type of the chord.

Options: major, minor, augmented, diminished, major7, minor7, sus2, sus4.

Beats list: the beats for the note items in this chord. Users need to give a beats list of certain length according to the chord type,

so either 3 beat items in the list for chord\_type [major,minor, augmented,diminished,sus2,sus4],

or 4 beat items in the list for chord\_type [major7, minor7].

The program will alert errors if failed to do so.

Example 1:

chord\_play() ->

[

{

"chord1",

1,

no\_delay,

"beep",

e3, minor,

['\_\_\_', '\_\_\_\_', '\_\_\_\_\_'],

no\_effect,

[{amplitude, 1.0}]

}

].

Example 2:

chord\_play() ->

[

{

"chord2",

1,

no\_delay,

"piano",

e3, minor7,

['\_\_\_', '\_\_\_\_', '\_\_\_\_\_', '\_\_'],

no\_effect,

[{amplitude, 1.0}]

},

{

"chord3",

1,

4.00,

"piano",

c4, major,

['\_\_\_', '\_\_\_\_', '\_\_\_\_\_'],

no\_effect,

[{amplitude, 1.0}]

}

].

## FUNCTION FOUR: sample\_pattern\_track()

This function is used to define the pattern tracks, in which you could play some samples according to a specific pattern.

Every tuple inside a '{}' is an independent track. For each tuple, you should define the following items:

name, mode, duration, delay, sample, pattern, effect, envelope.

Please refer to the syntax in the pitch\_based\_track() for the common items.

Sample: the sample you want to use. sample name needs to be surrounded by double quotes "".

Options:

"bass\_dnb\_f",

"drum\_heavy\_kick", "drum\_snare\_hard",

"guit\_harmonics","guit\_e\_fifths",

"ambi\_piano","perc\_till"

"loop\_safari","loop\_mehackit1","loop\_electric",

"table\_na\_o".

Pattern: the pattern for playing this sample. x means that play for 1 beat, o means that rest for 1 beat

Eg:

[x, o, x, o],

[x,x,x,x,x,x],

[x,o,o].

Example 1:

sample\_pattern\_track() ->

[

{"sample1", 1, 2.1, no\_delay,"drum\_heavy\_kick", [x, o, o], echo, [{amplitude, 0.5}] },

{"sample2", 1, 4.1, 4.25,"drum\_snare\_hard", [o, x, o, x], no\_effect, no\_envelope},

{"sample3", loop, 5.1, 4.25,"guit\_e\_fifths", [x, x, x, x, x], ping\_pong, no\_envelope},

{"sample4", 1, 2.0, 4.25,"perc\_till", [x, o, o, o], wobble, [{amplitude, 0.6}] }

].

Example 2:

sample\_pattern\_track() ->

[

{"sample1", 1, 3.1, no\_delay,"guit\_harmonics", [x, o, x, o], echo, [{amplitude, 0.5}] },

{"sample2", 1, 3.6, no\_delay,"drum\_snare\_hard", [o,x,x], no\_effect, no\_envelope},

{"sample3", 1, 0.8, no\_delay,"loop\_compus", [x], ping\_pong, no\_envelope},

].

## FUNCTION FIVE: rubato\_pattern\_track()

This function is used to define the rubato pattern tracks. This kind of track provides transitions from a given initial duration to a given end duration according to a sequence of xo events.

If the first event is short and the last is long, then the result will be a rallentando. Like:

\_ \_\_ \_\_\_ \_\_\_\_ \_\_\_\_\_ \_\_\_\_\_\_\_ \_\_\_\_\_\_\_\_\_

If the first event is long and the last is short, we get an accelerando. Like:

\_\_\_\_\_\_\_\_\_ \_\_\_\_\_\_\_\_ \_\_\_\_\_\_\_ \_\_\_\_\_ \_\_\_\_ \_\_\_ \_\_ \_

Every tuple inside a '{}' is an independent track. In each tuple, you should define the following items:

name, mode, first, last, delay, sample, pattern, effect, envelope.

Please refer to the syntax in the sample\_pattern\_track() for the common items.

First: the beat duration for the first beat in the xo sequences

Last: the beat duration for the last beat in the xo sequences

Example 1:

rubato\_pattern\_track() ->

[

{"rubato1", 1, '\_\_\_\_', '\_', no\_delay, "drum\_heavy\_kick", [x, x, x, x, x ,x ], no\_effect, [{amplitude, 1.0}] }

].

Example 2:

[

{"rubato1", loop, '\_', '\_\_\_\_\_', 6.00, "drum\_heavy\_kick", [x, x, x, x, x, x, x, x, x, x, x, x ,x ], no\_effect, [{amplitude, 1.0}] },

{"rubato2", 1, '\_\_\_\_\_', '\_', no\_delay, "drum\_heavy\_kick", [x, o, x, x, o, x, x ,x ], no\_effect, [{amplitude, 1.0}] }

].

## FUNCTION SIX: choose\_play()

This function is used to define the choose based tracks. In this kind of track, the program could play some random music by randomly choosing items from the note and beat lists that users give.

Every tuple inside a '{}' is an independent track. In each tuple, you should define the following items:

name, mode, delay, instrument, noteList, beatList.

Please refer to the syntax in the pitch\_based\_track() for the common items.

noteList: a list of notes. In each iteration of the loop, the program will randomly choose one note from the list to play.

beatList: a list of beats. In each iteration of the loop, the program will randomly choose one beat from the list to play.

In the following example 1, we have 5 iterations. The possible results for note and beat combinations could be:

Iteration 1: {eb4, '\_\_\_\_\_'}

Iteration 2: {gb4, '\_'}

Iteration 3: {eb4, '\_\_\_\_\_'}

Iteration 4: {65.89, '\_\_'}

Iteration 5: {gb4, '\_'}

Example 1:

choose\_play() ->

[

{

"choose1",

5,

4.00,

"piano",

[eb4 , 65.89, gb4],

['\_', '\_\_\_\_\_', '\_\_\_\_\_','\_\_']

}

].

Example 2:

choose\_play() ->

[

{

"choose1",

loop,

no\_delay,

"prophet",

[d5, e5, 78.59],

['\_\_', '\_\_\_']

},

{

"choose2",

6,

10.00,

"piano",

[60.25, 65.89, 78.59],

['\_\_', '\_\_\_', '\_\_\_\_\_','\_','\_\_\_']

}

].