tones Documentation

Release v0.01

Erik Nyquist

Nov 30, 2020

Contents

1 tones **3**
1.1 tones package 3

2 Indices and tables **9**

Python Module Index **11**

Index **13**

Contents:

CHAPTER 1

tones

1.1 tones package

1.1.1 Submodules

1.1.2 tones.mixer module

class tones.mixer.**Mixer**(*sample_rate=44100*, *amplitude=0.5*)

Bases: object

Represents multiple tracks that can be summed together into a single list of samples

add_note(*trackname*, *note='a'*, *octave=4*, *duration=1.0*, *endnote=None*, *endoctave=None*, *attack=None*, *decay=None*, *amplitude=1.0*, *vibrato_frequency=None*, *vibrato_variance=None*)

Same as 'add_tone', except the pitch can be specified as a standard musical note, e.g. "c#"

Parameters

- **trackname** – track identifier, track to add tone to
- **note** (*str*) – musical note. Must be a single character from a-g (non case-sensitive), followed by an optional sharp ('#') or flat ('b') character
- **endnote** (*str*) – If not None, the tone frequency will change between 'note' and 'endnote' in increments of 1ms over all samples
- **octave** (*int*) – note octave from 0-8
- **endoctave** (*int*) – octave for the note specified by endnote
- **duration** (*float*) – tone duration in seconds
- **attack** (*float*) – tone attack in seconds. Overrides the track's attack setting
- **decay** (*float*) – tone decay in seconds. Overrides the track's decay setting

- **vibrato_frequency** (*float*) – tone vibrato frequency in Hz. Overrides the track's vibrato frequency setting
- **vibrato_variance** (*float*) – tone vibrato variance in Hz. Overrides the track's vibrato variance setting
- **amplitude** (*float*) – Tone amplitude, where 1.0 is the max. sample value and 0.0 is total silence

add_notes (*trackname*, *notelist*)

Create multiple notes and add the samples for each tone in order to a track

Parameters

- **trackname** – track identifier
- **notelist** – list of tuples, where each tuple contains arguments for a single Mixer.add_note invocation

add_samples (*trackname*, *samples*)

Adds samples to a track

Parameters

- **trackname** – track identifier, track to add samples to
- **samples** (*[float]*) – samples to add

add_silence (*trackname*, *duration=1.0*)

Adds silence to a track

Parameters

- **trackname** – track identifier, track to add silence to
- **duration** (*float*) – silence duration in seconds

add_tone (*trackname*, *frequency=440.0*, *duration=1.0*, *endfrequency=None*, *attack=None*, *decay=None*, *amplitude=1.0*, *vibrato_frequency=None*, *vibrato_variance=None*)

Create a tone and add the samples to a track

Parameters

- **trackname** – track identifier, track to add tone to
- **frequency** (*float*) – tone frequency
- **duration** (*float*) – tone duration in seconds
- **attack** (*float*) – tone attack in seconds. Overrides the track's attack setting
- **decay** (*float*) – tone decay in seconds. Overrides the track's decay setting
- **vibrato_frequency** (*float*) – tone vibrato frequency in Hz. Overrides the track's vibrato frequency setting
- **vibrato_variance** (*float*) – tone vibrato variance in Hz. Overrides the track's vibrato variance setting
- **amplitude** (*float*) – Tone amplitude, where 1.0 is the max. sample value and 0.0 is total silence

add_tones (*tonelist*)

Create multiple tones and add the samples for each tone in order to a track

Parameters **tonelist** – list of tuples, where each tuple contains arguments for a single Mixer.add_tone invocation

create_track(*trackname*, **args*, ***kwargs*)
Creates a Tone track

Parameters

- **trackname** – unique identifier for track. Can be any hashable type.
- **args** – arguments for Track constructor
- **kwargs** – keyword arguments for Track constructor

get_attack(*trackname*)
Get the tone attack for a track

Parameters **trackname** – track identifier

Returns tone attack

Return type float

get_decay(*trackname*)
Get the tone decay for a track

Parameters **trackname** – track identifier

Returns tone decay

Return type float

get_vibrato_frequency(*trackname*)
Get the vibrato frequency for a track

Parameters **trackname** – track identifier

Returns vibrato frequency in Hz

Return type float

get_vibrato_variance(*trackname*)
Get the vibrato variance for a track

Parameters **trackname** – track identifier

Returns vibrato variance in Hz

Return type float

get_wavetype(*trackname*)
Get the waveform type for a track

Parameters **trackname** – track identifier

Returns track waveform type

Return type int

mix()
Mixes all tracks into a single stream of samples

Returns mixed samples

Return type *Samples*

sample_data()
Mixes all trackes down into a single stream of samples and returns the samples as a 16-bit PCM stream packed into a string

set_attack (*trackname*, *attack*)
Set the tone attack for a track. This attack will be applied to all tones added to this track.

Parameters

- **trackname** – track identifier
- **attack** (*float*) – attack time in seconds

set_decay (*trackname*, *decay*)
Set tone decay for a track. This decay will be applied to all tones added to this track

Parameters

- **trackname** – track identifier
- **decay** (*float*) – decay time in seconds

set_vibrato_frequency (*trackname*, *frequency*)
Set vibrato frequency for a track. This vibrato frequency will be applied to all tones added to this track

Parameters

- **trackname** – track identifier
- **frequency** (*float*) – vibrato frequency in Hz

set_vibrato_variance (*trackname*, *variance*)
Set vibrato variance for a track. The variance represents the full range that the highest and lowest points of the vibrato will reach, in Hz; for example, a tone at 440Hz with a vibrato variance of 20hz would oscillate between 450Hz and 430Hz. This vibrato variance will be applied to all tones added to this track

Parameters

- **trackname** – track identifier
- **variance** (*float*) – vibrato variance in Hz

set_wavetype (*trackname*, *wavetype*)
Sets the waveform type for a track

Parameters

- **trackname** – track identifier, track to set wavetype for
- **wavetype** (*int*) – waveform type

write_wav (*filename*, *sampledata=None*)
Mixes all tracks into a single stream of samples and writes to a .wav audio file

Parameters **filename** (*str*) – name of file to write

class tones.mixer.**Track** (*wavetype=0*, *attack=None*, *decay=None*, *vibrato_frequency=None*, *vibrato_variance=15.0*)
Bases: object

Represents a single track in a Mixer

append_samples (*samples*)
Append samples to this track. Samples should be in the range -1.0 to 1.0

Parameters **samples** (*[float]*) – samples to append

1.1.3 tones.tone module

class tones.tone.**Samples**

Bases: `list`

Extension of list class with methods useful for manipulating audio samples

serialize()

Serializes all samples

Returns serialized samples

Return type bytes

class tones.tone.**Tone**(*rate*, *amplitude*, *wavetype*)

Bases: `object`

Represents a fixed monophonic tone

samples(*num*, *frequency*, *endfrequency=None*, *attack=0.05*, *decay=0.05*, *phase=0.0*, *vphase=0.0*, *vibrato_frequency=None*, *vibrato_variance=20.0*)

Generate tone for a specific number of samples

Parameters

- **num** (*int*) – number of samples to generate
- **frequency** (*float*) – tone frequency in Hz
- **endfrequency** (*float*) – If not None, the tone frequency will change between 'frequency' and 'endfrequency' in increments of 1ms over all samples
- **attack** (*float*) – tone attack in seconds
- **decay** (*float*) – tone decay in seconds
- **phase** (*float*) – starting phase of generated tone in radians
- **vphase** (*float*) – starting phase of vibrato in radians
- **vibrato_frequency** (*float*) – vibrato frequency in Hz
- **vibrato_variance** (*float*) – vibrato variance in Hz

Returns samples in the range of -1.0 to 1.0, tone phase, vibrato phase

Return type tuple of the form (samples, phase, vibrato_phase)

1.1.4 Module contents

CHAPTER 2

Indices and tables

- genindex
- modindex
- search

Python Module Index

t

tones, 7
tones.mixer, 3
tones.tone, 7

Index

A

`add_note()` (*tones.mixer.Mixer method*), 3
`add_notes()` (*tones.mixer.Mixer method*), 4
`add_samples()` (*tones.mixer.Mixer method*), 4
`add_silence()` (*tones.mixer.Mixer method*), 4
`add_tone()` (*tones.mixer.Mixer method*), 4
`add_tones()` (*tones.mixer.Mixer method*), 4
`append_samples()` (*tones.mixer.Track method*), 6

C

`create_track()` (*tones.mixer.Mixer method*), 4

G

`get_attack()` (*tones.mixer.Mixer method*), 5
`get_decay()` (*tones.mixer.Mixer method*), 5
`get_vibrato_frequency()` (*tones.mixer.Mixer method*), 5
`get_vibrato_variance()` (*tones.mixer.Mixer method*), 5
`get_wavetype()` (*tones.mixer.Mixer method*), 5

M

`mix()` (*tones.mixer.Mixer method*), 5
`Mixer` (*class in tones.mixer*), 3

S

`sample_data()` (*tones.mixer.Mixer method*), 5
`Samples` (*class in tones.tone*), 7
`samples()` (*tones.tone.Tone method*), 7
`serialize()` (*tones.tone.Samples method*), 7
`set_attack()` (*tones.mixer.Mixer method*), 5
`set_decay()` (*tones.mixer.Mixer method*), 6
`set_vibrato_frequency()` (*tones.mixer.Mixer method*), 6
`set_vibrato_variance()` (*tones.mixer.Mixer method*), 6
`set_wavetype()` (*tones.mixer.Mixer method*), 6

T

`Tone` (*class in tones.tone*), 7
`tones` (*module*), 7
`tones.mixer` (*module*), 3
`tones.tone` (*module*), 7
`Track` (*class in tones.mixer*), 6

W

`write_wav()` (*tones.mixer.Mixer method*), 6