Synthesizer Documentation

Author

Jun 22, 2022

Contents:

1 synthesizer package **1**
1.1 Submodules . . . 1
1.2 synthesizer.player module . . . 1
1.3 synthesizer.synthesizer module . . . 1
1.4 synthesizer.writer module . . . 3
1.5 Module contents . . . 3

2 Indices and tables **5**

Python Module Index **7**

Index **9**

CHAPTER 1

synthesizer package

1.1 Submodules

1.2 synthesizer.player module

class synthesizer.player.**Player**(*rate=44100*)

Bases: object

enumerate_device()

open_stream(*device_name=None*, *device_index=-1*)

open audio output stream

if neither device_name nor device_index is specified, default output device will be opened.

Parameters

- **device_name** (*str*) – part of device name (ex: hw:0,0)
- **device_index** (*int*) – index of device

play_wave(*wave*)

play normalized wave

Parameters **wave** (*numpy.array*) – normalized wave

1.3 synthesizer.synthesizer module

class synthesizer.synthesizer.**Oscillator**(*waveform*, *volume*, *freq_transpose=1.0*)

Bases: object

Virtual oscillator object

Parameters

- **_waveform** (Waveform) – waveform of oscillator
- **_volume** (*float*) – amplitude of generated wave (0.1 - 1.0)
- **_freq_transpose** (*float*) – transpose of frequency (for sub oscillator)

static gen_triang(*t*, *width=0.5*)

generate_wave(*phases*)

volume

class synthesizer.synthesizer.**Synthesizer**(*osc1_waveform=<Waveform.sine: 'sine'>, osc1_volume=1.0, use_osc2=False, osc2_waveform=<Waveform.sine: 'sine'>, osc2_volume=0.3, osc2_freq_transpose=2.0, rate=44100*)

Bases: object

Virtual analog synthesizer object

Parameters

- **_osc1** (Oscillator) – main virtual oscillator
- **_osc2** (*(Oscillator|None)*) – sub virtual oscillator

generate_chord(*freqs*, *length*)

generate wave consists of multiple frequencies

Parameters

- **freqs** (*list[float]*) – list of frequencies or scales
- **length** – legnth of wave (seconds)

Return type numpy.array

Returns normalized wave

generate_constant_wave(*frequency*, *length*)

generate wave with constant frequency

Parameters

- **frequency** (*(float|str)*) – frequency or scale of wave
- **length** (*float*) – length of wave (seconds)

Return type numpy.array

Returns normalized wave

class synthesizer.synthesizer.**Waveform**

Bases: enum.Enum

sawtooth = 'sawtooth'

sine = 'sine'

square = 'square'

triangle = 'triangle'

1.4 synthesizer.writer module

class synthesizer.writer.**Writer**(*rate=44100*)

Bases: object

write_wave(*file_path*, *wave*)

write wave to a wav file

Parameters

- **file_path** (*str*) – relative / absolute path of wave file
- **wave** (*numpy.array*) – normalized wave

write_waves(*file_path*, **waves*)

write waves to a wav file

Parameters

- **file_path** (*str*) – relative / absolute path of wave file
- **waves** (*list[numpy.array]*) – list of normalized waves

1.5 Module contents

CHAPTER 2

Indices and tables

- genindex
- modindex
- search

Python Module Index

S

synthesizer, 3
synthesizer.player, 1
synthesizer.synthesizer, 1
synthesizer.writer, 3

Index

E

`enumerate_device()` (*synthesizer.player.Player method*), 1

G

`gen_triang()` (*synthesizer.synthesizer.Oscillator static method*), 2
`generate_chord()` (*synthesizer.synthesizer.Synthesizer method*), 2
`generate_constant_wave()` (*synthesizer.synthesizer.Synthesizer method*), 2
`generate_wave()` (*synthesizer.synthesizer.Oscillator method*), 2

O

`open_stream()` (*synthesizer.player.Player method*), 1
`Oscillator` (*class in synthesizer.synthesizer*), 1

P

`play_wave()` (*synthesizer.player.Player method*), 1
`Player` (*class in synthesizer.player*), 1

S

`sawtooth` (*synthesizer.synthesizer.Waveform attribute*), 2
`sine` (*synthesizer.synthesizer.Waveform attribute*), 2
`square` (*synthesizer.synthesizer.Waveform attribute*), 2
`Synthesizer` (*class in synthesizer.synthesizer*), 2
`synthesizer` (*module*), 3
`synthesizer.player` (*module*), 1
`synthesizer.synthesizer` (*module*), 1
`synthesizer.writer` (*module*), 3

T

`triangle` (*synthesizer.synthesizer.Waveform attribute*), 2

V

`volume` (*synthesizer.synthesizer.Oscillator attribute*), 2

W

`Waveform` (*class in synthesizer.synthesizer*), 2
`write_wave()` (*synthesizer.writer.Writer method*), 3
`write_waves()` (*synthesizer.writer.Writer method*), 3
`Writer` (*class in synthesizer.writer*), 3