
Adafruit WAVEFORM Library Documentation

Release 1.0

Scott Shawcroft

Aug 25, 2018

Contents

1	Dependencies	3
2	Usage Example	5
3	Contributing	7
4	Building locally	9
4.1	Sphinx documentation	9
5	Table of Contents	11
5.1	Simple tests	11
5.2	adafruit_waveform.sine	12
5.3	adafruit_waveform.square	12
6	Indices and tables	13
	Python Module Index	15

This library generates simple waveforms that can be used to generate different type of audio signals.

CHAPTER 1

Dependencies

This driver depends on:

- [Adafruit CircuitPython](#)

Please ensure all dependencies are available on the CircuitPython filesystem. This is easily achieved by downloading the [Adafruit library and driver bundle](#).

CHAPTER 2

Usage Example

This example generates one wavelength of a 440hz sine wave when played at 16 kilosamples per second:

```
from adafruit_waveform import sine
wave = sine.sine_wave(16000, 440)
```


CHAPTER 3

Contributing

Contributions are welcome! Please read our [Code of Conduct](#) before contributing to help this project stay welcoming.

CHAPTER 4

Building locally

To build this library locally you'll need to install the `circuitpython-build-tools` package.

```
python3 -m venv .env
source .env/bin/activate
pip install circuitpython-build-tools
```

Once installed, make sure you are in the virtual environment:

```
source .env/bin/activate
```

Then run the build:

```
circuitpython-build-bundles --filename_prefix adafruit-circuitpython-waveform --
↳library_location .
```

4.1 Sphinx documentation

Sphinx is used to build the documentation based on rST files and comments in the code. First, install dependencies (feel free to reuse the virtual environment from above):

```
python3 -m venv .env
source .env/bin/activate
pip install Sphinx sphinx-rtd-theme
```

Now, once you have the virtual environment activated:

```
cd docs
sphinx-build -E -W -b html . _build/html
```

This will output the documentation to `docs/_build/html`. Open the `index.html` in your browser to view them. It will also (due to `-W`) error out on any warning like Travis will. This is a good way to locally verify it will pass.

5.1 Simple tests

Ensure your device works with these simple tests.

Listing 1: examples/sine_demo.py

```
1  """
2  'sine_demo.py'.
3
4  =====
5  toggles the builtin LED using a sine wave
6  """
7  import time
8  import board
9  import digitalio
10 from adafruit_waveform import sine
11
12 LED = digitalio.DigitalInOut(board.D13)
13 LED.switch_to_output()
14
15 SINE_SAMPLE = sine.sine_wave(150, 50)
16
17 while True:
18     for i in range(len(SINE_SAMPLE)):
19         LED.value = i
20         print(LED.value)
21         time.sleep(0.50)
```

Listing 2: examples/square_demo.py

```
1  """
2  'square_demo.py'.
3
```

(continues on next page)

(continued from previous page)

```
4  =====
5  toggles the builtin LED using a square wave
6  """
7  import time
8  import digitalio
9  import board
10 from adafruit_waveform import square
11
12 LED = digitalio.DigitalInOut(board.D13)
13 LED.switch_to_output()
14 SAMPLE_SQUARE = square.square_wave(2)
15
16 while True:
17     for i in range(len(SAMPLE_SQUARE)):
18         LED.value = i
19         print(LED.value)
20         time.sleep(0.5)
```

5.2 adafruit_waveform.sine

This library generates sine waveforms that can be used to generate sine audio signals.

- Author(s): Scott Shawcroft

adafruit_waveform.sine.**sine_wave**(*sample_frequency*, *pitch*)

Generate a single sine wav cycle at the given sampling frequency and pitch.

5.3 adafruit_waveform.square

This library generates square waveforms that can be used to generate square audio signals.

- Author(s): Scott Shawcroft, BrentRu

adafruit_waveform.square.**square_wave**(*sample_length=2*)

Generate a single square wave of sample_length size

CHAPTER 6

Indices and tables

- `genindex`
- `modindex`
- `search`

a

`adafruit_waveform.sine`, [12](#)

`adafruit_waveform.square`, [12](#)

A

`adafruit_waveform.sine` (module), [12](#)
`adafruit_waveform.square` (module), [12](#)

S

`sine_wave()` (in module `adafruit_waveform.sine`), [12](#)
`square_wave()` (in module `adafruit_waveform.square`), [12](#)