# Twelve Tone

*Release 0.4.2*

**Apr 12, 2021**

# Contents


**1 Overview** **1**
- 1.1 Installation . . . 2
- 1.2 Quick Start . . . 2
- 1.3 Documentation . . . 2
- 1.4 Development . . . 2

**2 Installation** **3**

**3 Usage** **5**

**4 Reference** **7**
- 4.1 twelve_tone . . . 7

**5 Contributing** **9**
- 5.1 Bug reports . . . 9
- 5.2 Documentation improvements . . . 9
- 5.3 Feature requests and feedback . . . 9
- 5.4 Development . . . 10

**6 Authors** **11**

**7 Changelog** **13**
- 7.1 0.4.2 (2021-03-11) . . . 13
- 7.2 0.4.1 (2019-12-31) . . . 13
- 7.3 0.4.0 (2018-7-08) . . . 13
- 7.4 0.3.0 (2018-7-04) . . . 13
- 7.5 0.2.1 (2016-8-27) . . . 13
- 7.6 0.2.0 (2016-8-27) . . . 14
- 7.7 0.1.0 (2016-8-20) . . . 14

**8 Indices and tables** **15**

**Python Module Index** **17**

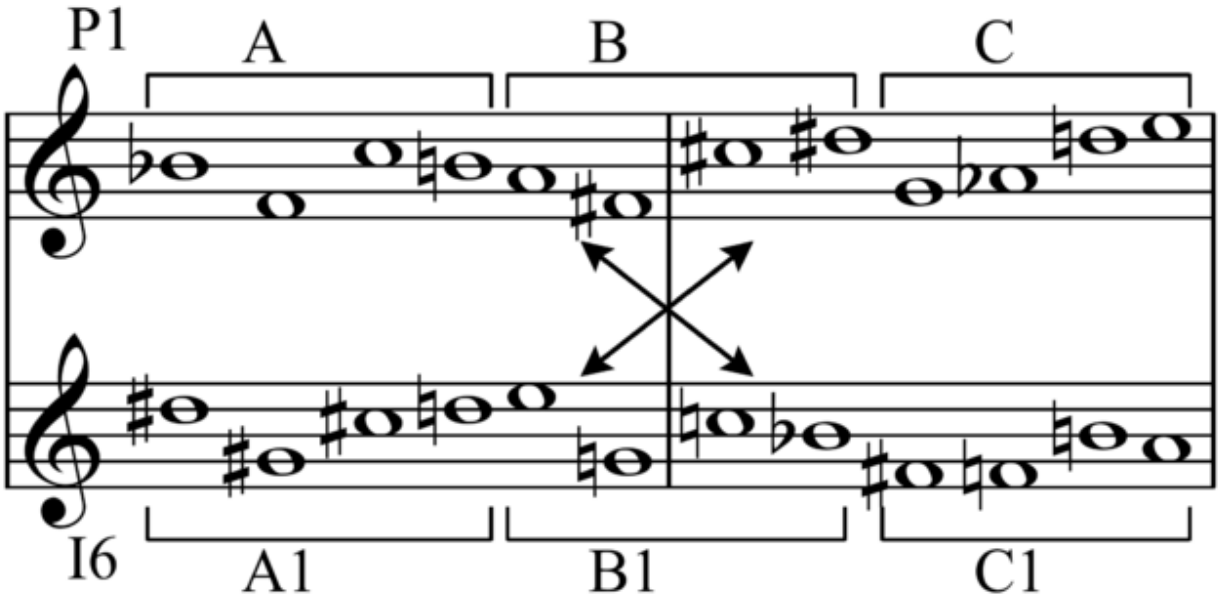
**Index** **19**

CHAPTER 1

# Overview

| | |
|---|---|
| docs | |
| tests | |
| package | |

Twelve-tone matrix to generate dodecaphonic melodies.




Following a process created by the composer Arnold Schoenberg, this library computes a matrix to create twelve-tone serialism melodies which compose each of the 12 semitones of the chromatic scale with equal importance.

- Save your compositions to MIDI
- Free software: BSD license

## 1.1 Installation

```
pip install twelve-tone
```

## 1.2 Quick Start

You can quickly generate a random twelve-tone melody with the CLI

```
$ twelve-tone
['C# / Db', 'A# / Bb', 'F', 'D', 'G# / Ab', 'D# / Eb', 'F# / Gb',
    'A', 'C', 'G', 'B', 'E']
```

Or you can use the following methods in a script:

```
>>> from twelve_tone.composer import Composer
>>> c = Composer()
>>> c.compose()
>>> c.get_melody()
['C# / Db', 'A# / Bb', 'F', 'D', 'G# / Ab', 'D# / Eb', 'F# / Gb',
    'A', 'C', 'G', 'B', 'E']
```

After you have composed a matrix of tone rows, you can save the composition to MIDI:

```
>>> c.compose()
>>> c.save_to_midi(filename='TWELVE_TONE.mid')
```

The new MIDI file will be created in your current working directory. If you do not specify a *filename* for your file, it will default to *example.mid*.

## 1.3 Documentation

https://python-twelve-tone.readthedocs.io/

## 1.4 Development

To run the all tests run:

```
tox
```

Note, to combine the coverage data from all the tox environments run:

| | |
|---|---|
| Windows | `set PYTEST_ADDOPTS=--cov-append`<br>`tox` |
| Other | `PYTEST_ADDOPTS=--cov-append tox` |

CHAPTER 2

# Installation

At the command line:

```
pip install twelve-tone
```

# CHAPTER 3

# Usage

To use Twelve Tone in a project:

```
import twelve_tone
```

You can quickly generate a random twelve-tone melody with the CLI

```
$ twelve-tone
['C# / Db', 'A# / Bb', 'F', 'D', 'G# / Ab', 'D# / Eb', 'F# / Gb',
    'A', 'C', 'G', 'B', 'E']
```

Or you can use the following methods in a script:

```
>>> from twelve_tone.composer import Composer
>>> c = Composer()
>>> c.compose()
>>> c.get_melody()
['C# / Db', 'A# / Bb', 'F', 'D', 'G# / Ab', 'D# / Eb', 'F# / Gb',
    'A', 'C', 'G', 'B', 'E']
```

After you have composed a matrix of tone rows, you can save the composition to MIDI:

```
>>> c.compose()
>>> c.save_to_midi(filename='TWELVE_TONE.mid')
```

You can even select either a specific row or column when generating a melody.

```
from twelve_tone.composer import Composer
c = Composer()
c.compose()
melody_row = c.get_melody(row=3)
melody_col = c.get_melody(column=7)
```

CHAPTER 4

# Reference

## 4.1 twelve_tone

CHAPTER 5

# Contributing

Contributions are welcome, and they are greatly appreciated! Every little bit helps, and credit will always be given.

## 5.1 Bug reports

When reporting a bug please include:

- Your operating system name and version.
- Any details about your local setup that might be helpful in troubleshooting.
- Detailed steps to reproduce the bug.

## 5.2 Documentation improvements

Twelve Tone could always use more documentation, whether as part of the official Twelve Tone docs, in docstrings, or even on the web in blog posts, articles, and such.

## 5.3 Feature requests and feedback

The best way to send feedback is to file an issue at https://github.com/accraze/python-twelve-tone/issues.

If you are proposing a feature:

- Explain in detail how it would work.
- Keep the scope as narrow as possible, to make it easier to implement.
- Remember that this is a volunteer-driven project, and that code contributions are welcome :)

## 5.4 Development

To set up *python-twelve-tone* for local development:

1. Fork python-twelve-tone (look for the "Fork" button).
2. Clone your fork locally:

```
git clone git@github.com:your_name_here/python-twelve-tone.git
```

3. Create a branch for local development:

```
git checkout -b name-of-your-bugfix-or-feature
```

   Now you can make your changes locally.

4. When you're done making changes, run all the checks, doc builder and spell checker with tox one command:

```
tox
```

5. Commit your changes and push your branch to GitHub:

```
git add .
git commit -m "Your detailed description of your changes."
git push origin name-of-your-bugfix-or-feature
```

6. Submit a pull request through the GitHub website.

### 5.4.1 Pull Request Guidelines

If you need some code review or feedback while you're developing the code just make the pull request.

For merging, you should:

1. Include passing tests (run `tox`)[1].
2. Update documentation when there's new API, functionality etc.
3. Add a note to `CHANGELOG.rst` about the changes.
4. Add yourself to `AUTHORS.rst`.

### 5.4.2 Tips

To run a subset of tests:

```
tox -e envname -- py.test -k test_myfeature
```

To run all the test environments in *parallel* (you need to `pip install detox`):

```
detox
```

[1] If you don't have all the necessary python versions available locally you can rely on Travis - it will run the tests for each change you add in the pull request.
It will be slower though ...

# CHAPTER 6

# Authors

- Andy Craze - https://github.com/accraze

CHAPTER 7

# Changelog

## 7.1 0.4.2 (2021-03-11)

### 0.4.1 (2021-3-11)

- requirements: added missing dependency click - (#22) - @jgarte

## 7.2 0.4.1 (2019-12-31)

### 0.4.1 (2019-12-31)

- composer: matrix should only hold values of type 'int' (#20)

## 7.3 0.4.0 (2018-7-08)

- composer: added/fixed column tonerow support

## 7.4 0.3.0 (2018-7-04)

- cli: added random melody generator command

## 7.5 0.2.1 (2016-8-27)

- build: added *miditime* to setup install requirements

## 7.6 0.2.0 (2016-8-27)

- composer: Added save to MIDI capability

## 7.7 0.1.0 (2016-8-20)

- First release on PyPI.

CHAPTER 8

# Indices and tables

- genindex
- modindex
- search

# Python Module Index

## t

twelve_tone, 7

# Index

## T

`twelve_tone` (*module*), 7