## Contents

<table>
<thead>
<tr>
<th>Section</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>1 Readme</td>
<td>3</td>
</tr>
<tr>
<td>1.1 DataSounds</td>
<td>3</td>
</tr>
<tr>
<td>2 Installation</td>
<td>5</td>
</tr>
<tr>
<td>2.1 Dependencies</td>
<td>5</td>
</tr>
<tr>
<td>3 Usage</td>
<td>7</td>
</tr>
<tr>
<td>3.1 Simple usage</td>
<td>7</td>
</tr>
<tr>
<td>3.2 Setting parameters</td>
<td>7</td>
</tr>
<tr>
<td>3.3 List of MIDI instruments</td>
<td>7</td>
</tr>
<tr>
<td>4 Contributing</td>
<td>11</td>
</tr>
<tr>
<td>4.1 Types of Contributions</td>
<td>11</td>
</tr>
<tr>
<td>4.2 Get Started!</td>
<td>12</td>
</tr>
<tr>
<td>4.3 Pull Request Guidelines</td>
<td>12</td>
</tr>
<tr>
<td>4.4 Tips</td>
<td>13</td>
</tr>
<tr>
<td>5 Credits</td>
<td>15</td>
</tr>
<tr>
<td>5.1 DataSounds development Lead</td>
<td>15</td>
</tr>
<tr>
<td>5.2 Contributors</td>
<td>15</td>
</tr>
<tr>
<td>6 History</td>
<td>17</td>
</tr>
<tr>
<td>6.1 1.2.0 (2013-12-22)</td>
<td>17</td>
</tr>
<tr>
<td>6.2 1.1.0 (2013-06-08)</td>
<td>17</td>
</tr>
<tr>
<td>6.3 0.1.0 (2013-03-15)</td>
<td>17</td>
</tr>
<tr>
<td>7 Indices and tables</td>
<td>19</td>
</tr>
</tbody>
</table>
Contents:
1.1 DataSounds

Get sounds from temporal series, or another sequencing data. Visit us at www.datasounds.org.


Installation

At the command line:

```
$ git clone http://github.com/DataSounds/DataSounds.git
$ cd DataSounds
$ python setup.py install
```

Or using pip (program to easily install Python packages), which dynamically access the Python Package Index PyPI.

```
$ pip install DataSounds
```

Or, with the controlled python ecosystem virtualenvwrapper. After install virtualenvwrapper, follow the instructions below:

```
$ mkvirtualenv DataSounds
$ workon DataSounds
(DataSounds) $ git clone http://github.com/DataSounds/DataSounds.git
(DataSounds) $ cd DataSounds
(DataSounds) $ python setup.py install
```

2.1 Dependencies

**Numpy** is a necessary package to use DataSounds.

Numpy can be installed using pip. If you use virtualenvwrapper, this could be done inside your virtual environment. Normally, **Numpy** is installed as a dependency of DataSounds and should work if it was successfully compiled.
3.1 Simple usage

Simple Python script to use DataSounds module to sonify dataset.

```python
from DataSounds.sounds import get_music, w2Midi
import numpy as np

data = np.random.rand(24)
music = get_music(data)
w2Midi('my_music_data', music)
```

w2Midi writes a .midi file inside the current directory. In this case my_music_data.midi will be saved on disk.

Play MIDI files inside Python is a little bit tricky, and it is possible using the compilation of Pygame library.

To not bring this dependency inside DataSounds, we try to use the system player. For Linux you could install timidity (apt-get install timidity) or use totem with fluid-soundfont-gm. For OS X users, we try to use the simple open or timidity, and for windows users you should install timidity.

DataSounds comes with default pre settings of get_music function, which may be changed accordingly the user preferences. The help function displays:

3.2 Setting parameters

Parameters can be changed if you want to use DataSounds to differentiate two or more series. This usage requires the difference of instruments and also the number of octaves of each series. For example:

This example shows how to use different instruments for each series (e.g. Acoustic Grand Piano for ts_a and Rock Organ for ts_b.

The usage of octaves can better perform distinction between values. We can correlate this parameter as colorbars, if you have a colorbar with 7 colors to represent more than 7 distinct values, your display may be masking some results. Same logical procedure can be adopted here.

3.3 List of MIDI instruments

Numbers at left side of instrument name are already written considering Python listing index (e.g. first number is $== 0$).
## Piano
- 0 Acoustic Grand Piano
- 1 Bright Acoustic Piano
- 2 Electric Grand Piano
- 3 Honky-tonk Piano
- 4 Electric Piano 1
- 5 Electric Piano 2
- 6 Harpsichord
- 7 Clavinet

## Chromatic Percussion
- 8 Celesta
- 9 Glockenspiel
- 10 Music Box
- 11 Vibraphone
- 12 Marimba
- 13 Xylophone
- 14 Tubular Bells
- 15 Dulcimer

## Organ
- 16 Drawbar Organ
- 17 Percussive Organ
- 18 Rock Organ
- 19 Church Organ
- 20 Reed Organ
- 21 Accordion
- 22 Harmonica
- 23 Tango Accordion

## Guitar
- 24 Acoustic Guitar(nylon)
- 25 Acoustic Guitar(steel)
- 26 Electric Guitar(jazz)
- 27 Electric Guitar(clean)
- 28 Electric Guitar(muted)
- 29 Overdriven Guitar
- 30 Distortion Guitar
- 31 Guitar Harmonics

## Bass
- 32 Acoustic Bass
- 33 Electric Bass (finger)
- 34 Electric Bass (pick)
- 35 Fretless Bass
- 36 Slap Bass 1
- 37 Slap Bass 2
- 38 Synth Bass 1
- 39 Synth Bass 2

## Strings
- 40 Violin
- 41 Viola
- 42 Cello
- 43 Contrabass
- 44 Tremolo String
- 45 Pizzicato Strings
- 46 Orchestral Harp
- 47 Timpani

## Ensemble
- 48 String Ensemble 1
- 49 String Ensemble 2
- 50 Synth Strings 1
- 51 Synth Strings 2
- 52 Choir Aahs
- 53 Voice Ooohs
- 54 Synth Choir
- 55 Orchestra Hit

## Brass
- 56 Trumpet
- 57 Trombone
- 58 Tuba
- 59 Muted Trumpet
- 60 French Horn
- 61 Brass Section
- 62 Synth Brass 1
- 63 Synth Brass 2

## Reed
- 64 Soprano Sax
- 65 Alto Sax
- 66 Tenor Sax
- 67 Baritone Sax
- 68 Oboe
- 69 English Horn
- 70 Bassoon
- 71 Clarinet
### 3.3. List of MIDI instruments

<table>
<thead>
<tr>
<th>Pipe</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>72 Piccolo</td>
<td>73 Flute</td>
</tr>
<tr>
<td>74 Recorder</td>
<td>75 Pan Flute</td>
</tr>
<tr>
<td>76 Blown bottle</td>
<td>77 Shakuhachi</td>
</tr>
<tr>
<td>78 Whistle</td>
<td>79 Ocarina</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Synth Lead</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>80 Lead 1 (square)</td>
<td>81 Lead 2 (sawtooth)</td>
</tr>
<tr>
<td>82 Lead 3 (calliope)</td>
<td>83 Lead 4 chiff</td>
</tr>
<tr>
<td>84 Lead 5 (charang)</td>
<td>85 Lead 6 (voice)</td>
</tr>
<tr>
<td>86 Lead 7 (fifths)</td>
<td>87 Lead 8 (bass + lead)</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Synth Pad</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>88 Pad 1 (new age)</td>
<td>89 Pad 2 (warm)</td>
</tr>
<tr>
<td>90 Pad 3 (polysynth)</td>
<td>91 Pad 4 (choir)</td>
</tr>
<tr>
<td>92 Pad 5 (bowed)</td>
<td>93 Pad 6 (metallic)</td>
</tr>
<tr>
<td>94 Pad 7 (halo)</td>
<td>95 Pad 8 (sweep)</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Synth Effects</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>96 FX 1 (rain)</td>
<td>97 FX 2 (soundtrack)</td>
</tr>
<tr>
<td>98 FX 3 (crystal)</td>
<td>99 FX 4 (atmosphere)</td>
</tr>
<tr>
<td>100 FX 5 (brightness)</td>
<td>101 FX 6 (goblins)</td>
</tr>
<tr>
<td>102 FX 7 (echoes)</td>
<td>103 FX 8 (sci-fi)</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Ethnic</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>104 Sitar</td>
<td>105 Banjo</td>
</tr>
<tr>
<td>106 Shamisen</td>
<td>107 Koto</td>
</tr>
<tr>
<td>108 Kalimba</td>
<td>109 Bagpipe</td>
</tr>
<tr>
<td>110 Fiddle</td>
<td>111 Shanai</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Percussive</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>112 Tinkle Bell</td>
<td>113 Agogo</td>
</tr>
<tr>
<td>114 Steel Drums</td>
<td>115 Woodblock</td>
</tr>
<tr>
<td>116 Taiko Drum</td>
<td>117 Melodic Tom</td>
</tr>
<tr>
<td>118 Synth Drum</td>
<td>119 Reverse Cymbal</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Sound effects</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>120 Guitar Fret Noise</td>
<td>121 Breath Noise</td>
</tr>
<tr>
<td>122 Seashore</td>
<td>123 Bird Tweet</td>
</tr>
<tr>
<td>124 Telephone Ring</td>
<td>125 Helicopter</td>
</tr>
<tr>
<td>126 Applause</td>
<td>127 Gunshot</td>
</tr>
</tbody>
</table>
Contributions are welcome, and they are greatly appreciated! Every little bit helps, and credit will always be given. You can contribute in many ways:

### 4.1 Types of Contributions

#### 4.1.1 Report Bugs


If you are 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.

#### 4.1.2 Fix Bugs

Look through the GitHub issues for bugs. Anything tagged with “bug” is open to whoever wants to implement it.

#### 4.1.3 Implement Features

Look through the GitHub issues for features. Anything tagged with “feature” is open to whoever wants to implement it.

#### 4.1.4 Write Documentation

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

#### 4.1.5 Submit Feedback

The best way to send feedback is to file an issue at https://github.com/DataSounds/DataSounds/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 contributions are welcome :)

4.2 Get Started!

Ready to contribute? Here’s how to set up DataSounds for local development.

1. Fork the DataSounds repo on GitHub.
2. Clone your fork locally:
   ```
   $ git clone git@github.com:your_name_here/DataSounds.git
   ```
3. Install your local copy into a virtualenv. Assuming you have virtualenvwrapper installed, this is how you set up your fork for local development:
   ```
   $ mkvirtualenv devDS
   $ cd DataSounds/
   $ python setup.py develop
   ```
4. Create a branch for local development:
   ```
   $ git checkout -b name-of-your-bugfix-or-feature
   ```
   Now you can make your changes locally.
5. When you’re done making changes, check that your changes pass through tests, including testing other Python versions with tox:
   ```
   $ python setup.py test
   $ tox
   ```
   To get the necessary packages just install it as follows inside yours virtualenv:
   ```
   $ pip install -r dev-requirements.txt
   ```
6. 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
   ```
7. Submit a pull request through the GitHub website.

4.3 Pull Request Guidelines

Before you submit a pull request, check that it meets these guidelines:

1. The pull request should include tests.
2. If the pull request adds functionality, the docs should be updated. Put your new functionality into a function with a docstring, and add the feature to the list in README.rst.
3. The pull request should work for Python 2.6, 2.7, and 3.3, and for PyPy. Check https://travis-ci.org/DataSounds/DataSounds/pull_requests and make sure that the tests pass for all supported Python versions.
4.4 Tips

To run a subset of tests:

```bash
$ python -m unittest tests.test_sounds.py
```
5.1 DataSounds development Lead

- Arnaldo Russo <arnaldo@datasounds.org>
- Luiz Irber <luiz@datasounds.org>

5.2 Contributors

None yet. Why not be the first?
History

6.1 1.2.0 (2013-12-22)

6.2 1.1.0 (2013-06-08)

6.3 0.1.0 (2013-03-15)

• First release on PyPI.
CHAPTER 7

Indices and tables

- genindex
- modindex
- search