guano-py Documentation

Release 1.0.12

David A. Riggs

Aug 09, 2018

Contents

This is the Python reference implementation for GUANO, the "Grand Unified Acoustic Notation Ontology", a universal metadata format for bat acoustic recordings. It includes a production-ready Python module with full support for reading and writing GUANO metadata, as well as several helpful commandline utilities. This is *Free Software* which may be used for *any* purpose.

For more information about GUANO metadata itself, including the format specification, see the GUANO project homepage: http://guano-md.org

Documentation for guano-py can be found at: http://guano-py.readthedocs.io

CHAPTER 1

Table of Contents

1.1 Installation

1.1.1 Requirements

- Python 2.7 or Python 3.3+

1.1.2 Installation

Download and install magically from the Python Package Index:

```
$> pip install -U guano
```

In addition to having the *guano Python module* available for use in your own software, you'll also have a small collection of *useful scripts* to use.

Alternately, you can clone the guano-py GitHub project and install locally in developer mode to hack on it yourself:

```
$> git clone https://github.com/riggsd/guano-py.git
$> cd guano-py
$> python setup.py develop
```

1.2 The *guano* Python Module

This is the Python reference implementation for reading and writing GUANO metadata.

GUANO is the "Grand Unified Acoustic Notation Ontology", an extensible metadata format for representing bat acoustics data.

Import this Python module as:

```
import guano
```

This module utilizes the Python `logging` framework for issuing warnings and debug messages. Application code may wish to enable logging with the `logging.basicConfig()` function.

class guano.**GuanoFile** (*filename=None*, *strict=True*)

An abstraction of a .WAV file with GUANO metadata.

A *GuanoFile* object behaves like a normal Python `dict`, where keys can either be well-known metadata keys, namespaced keys, or a tuple of (namespace, key).

Well-known keys will have their values coerced into the correct data type. The parser may be configured to coerce new namespaced keys with the `register()` function.

Example usage:

```
gfile = GuanoFile('myfile.wav')
print gfile['GUANO|Version']
>>> '1.0'
gfile['Species Manual ID'] = 'Mylu'
gfile['Note'] = 'I love GUANO!'
gfile.write()
```

Though reading, writing, and editing .WAV files is the target usage, this class may also be used independent from the .WAV file format. GUANO metadata can be written into an Anabat-format file or to a sidecar file, for example, by populating a *GuanoFile* object and then using the `serialize()` method to produce correctly formatted UTF-8 encoded metadata.

Variables

- **filename** (*str*) – path to the file which this object represents, or *None* if a "new" file
- **strict_mode** (*bool*) – whether the GUANO parser is configured for strict or lenient parsing
- ***wav_data*** (*bytes*) – the *data* subchunk of a .WAV file consisting of its actual audio data, lazily-loaded and cached for performance
- **wav_params** (*wavparams*) – namedtuple of .WAV parameters (nchannels, sampwidth, framerate, nframes, comptype, compname)

classmethod from_string (*metadata_str*, **args*, ***kwargs*)

Create a `GuanoFile` instance from a GUANO metadata string

Parameters **metadata_str** – a string (or string-like buffer) of GUANO metadata

Return type *GuanoFile*

get_namespaces ()

Get list of all namespaces represented by this metadata. This includes the 'GUANO' namespace, and the '' (empty string) namespace for well-known fields.

items (*namespace=None*)

Iterate over (key, value) for entire metadata or for specified namespace of fields

items_namespaced ()

Iterate over (namespace, key, value) for entire metadata

classmethod register (*namespace*, *keys*, *coerce_function*, *serialize_function=<type 'str'>*)

Configure the GUANO parser to recognize new namespaced keys.

Parameters

- **namespace** – vendor namespace which the keys belong to
- **keys** – a key or sequence of keys under the specified vendor namespace
- **coerce_function** (*callable*) – a function for coercing the UTF-8 value to any desired data type
- **serialize_function** (*callable*) – an optional function for serializing the value to UTF-8 string

serialize(*pad='\n'*)
Serialize the GUANO metadata as UTF-8 encoded bytes

to_string()
Represent the GUANO metadata as a Unicode string

wav_data
Actual audio data from the wav *data* chunk. Lazily loaded and cached.

well_known_items()
Iterate over (key, value) for all the well-known (defined) fields

write(*make_backup=True*)
Write the GUANO .WAV file to disk.

Parameters **make_backup** (*bool*) – create a backup file copy before writing changes or not (default: True); backups will be saved to a folder named *GUANO_BACKUP*

Raises **ValueError** – if this *GuanoFile* doesn't represent a valid .WAV by having appropriate values for *self.wav_params* (see `wave.Wave_write.setparams()`) and *self.wav_data* (see `wave.Wave_write.writeframes()`)

1.3 Example API Use

```
from guano import GuanoFile

# load a .WAV file with (or without) GUANO metadata
g = GuanoFile('test.wav')

# get and set metadata values like a Python dict
print g['GUANO|Version']
>>> 1.0

print g['Make'], g['Model']
>>> 'Pettersson', 'D500X'

g['Species Manual ID'] = 'Myso'

g['Note'] = 'I love GUANO!'

# namespaced fields can be specified separately or pipe-delimited
print g['PET', 'Gain'], g['PET|Gain']
>>> 80, 80

g['SB|Consensus'] = 'Epfu'
g['SB', 'Consensus'] = 'Epfu'

# print all the metadata values
```

(continues on next page)

(continued from previous page)

```
for key, value in g.items():
    print '%s: %s' % (key, value)

# write the updated .WAV file back to disk
g.write()

# have some GUANO metadata from some other source? load it from a string
g = GuanoFile.from_string('GUANO|Version:1.0\nTags:voucher,hand-release')

# write GUANO metadata somewhere else, say an Anabat file or text file
with open('sidecar_file.guano', 'wb') as outfile:
    outfile.write( g.serialize() )

# teach the parser to recognize custom metadata fields
GuanoFile.register('Anabat', ['Humidity', 'Temperature'], float)
GuanoFile.register('SB', 'Thumbnail Image', guano.base64decode)
```

1.4 Utilities

The guano-py Python module includes several helpful commandline utilities for working with GUANO metadata. When you install guano-py with *pip install guano*, the following scripts will then be callable from the commandline.

1.4.1 guano_dump.py

Print the GUANO metadata found in a file or files.

usage:

```
$> guano_dump.py WAVFILE...
```

1.4.2 guano_edit.py

guano_edit.py - Manipulate GUANO metadata of individual files or in bulk.

Specify GUANO fields and values, followed by a list of files which the changes should be applied to. The values of existing fields may be used by specifying the field as *${Fieldname}*.

Be careful to properly escape values for your shell commandline, especially if using value templates! Study the examples below.

Examples:

```
# Add NABat grid cell to all recordings under the `foo` directory
$> guano_edit.py  "NABat|Grid Cell ID: 45678"  ~/bat_calls/foo/

# Append additional text to the end of the existing Note text
$> guano_edit.py  'Note: ${Note} Recorded by Dave.'  EPFU_refcall.wav
```

TODO::

- Ensure that we persist all RIFF chunks
- Add support for adding GUANO metadata to "new" files

- Support Anabat files

1.4.3 d500x2guano.py

Convert files with raw D500X metadata to use GUANO metadata instead.

usage:

```
$> d500x2guano.py WAVFILE...
```

1.4.4 sb2guano.py

Convert files with SonoBat-format metadata to use GUANO metadata instead.

usage:

```
$> sb2guano.py WAVFILE...
```

1.4.5 wamd2guano.py

Convert Wildlife Acoustics WAMD metadata files to use GUANO metadata instead.

usage:

```
$> wamd2guano.py WAVFILE...
```

1.4.6 batlogger2guano.py

Convert files from the Elekon BatLogger to use GUANO metadata instead.

usage:

```
$> batlogger2guano.py WAVFILE...
```

1.4.7 disperse.py

“Disperse” files by moving them into folders according to their species label.

The *Species Manual ID* field will be preferred over *Species Auto ID*.

usage:

```
$> disperse.py [--copy] ROOTDIR
```

1.5 The MIT License (MIT)

1.6 Changelog

1.0.12

2018-08-08

- Allow setting *GuanoFile.wav_data*, and fix bug in *d500x2guano.py* script which strips Pettersson metadata out of the *data* chunk

1.6.1 1.0.11

2017-12-14

- Version bump to reflect that this is production-quality code

1.6.2 0.0.11

2017-12-14

- Huge improvement in read speed and memory usage by lazily-loading *GuanoFile.wav_data* and by *not* using mmap for file access
- Add support for deleting fields as *del gfile['Species Manual ID']*
- *disperse.py*: add *–copy* option to copy rather than move files

1.6.3 0.0.10

2017-09-30

- Add *guano_edit.py* util for adding and changing files' GUANO metadata
- Add strict/lenient parsing option to *GuanoFile* constructor
- Use Python's *logging* framework

1.6.4 0.0.9

2017-07-07

- Treat the *GUANO|Version* value as a string rather than numeric value

1.6.5 0.0.8

2017-05-19

- Try to recover metadata which is incorrectly encoded (UTF-8 is required, we try to fall back to latin-1)

1.6.6 0.0.7

2017-04-29

- Python 3 support
- User manual

1.6.7 0.0.6

2017-04-27

- Critical bugfix for parsing some UTF-8 text
- Additional resilience when parsing bad or non-conforming GUANO metadata
- Seamlessly escape/unescape multiline string *Note* field
- Addition of *batlogger2guano.py* utility script for converting Elekon Batlogger files to use GUANO instead

1.6.8 0.0.5

2017-03-18

- Add *GuanoFile()* constructor without filename for creating metadata instance which isn't tied to an underlying .WAV file

CHAPTER 2

Indices and tables

- genindex
- modindex
- search

Python Module Index

Index