Morse Talk Documentation

Release 0.1.1

Himanshu Mishra

December 25, 2016

Contents

1 Overview **3**
- 1.1 Goals . . . 3
- 1.2 The Python programming language . . . 3
- 1.3 Free software . . . 3
- 1.4 History . . . 3

2 Download **5**
- 2.1 Software . . . 5

3 Installing **7**
- 3.1 Quick install . . . 7
- 3.2 Requirements . . . 7
- 3.3 Optional packages . . . 7

4 Tutorial **9**
- 4.1 Encode a message . . . 9
- 4.2 Decode a message . . . 9

5 Testing **11**
- 5.1 Requirements for testing . . . 11
- 5.2 Testing a source distribution . . . 11
- 5.3 Testing an installed package . . . 11
- 5.4 Testing for developers . . . 11

6 Indices and tables **13**

Contents:

CHAPTER 1

Overview

Morse Talk is a Python language software package written for interpretation of Morse Code and utilizing it in all possible way. Currently it focuses in encoding and decoding and generating sounds for the codes.

1.1 Goals

NetworkX is intended to provide

- Encoding and decoding Morse Code in default and binary encription,
- More types of Encription,
- Converting Morse Codes into sound signals,
- Generating wave forms for the encoded signals.

1.2 The Python programming language

Python is a powerful programming language that allows simple and flexible ways to deal with patterns and dictionaries. Python has a vibrant and growing ecosystem of packages that Morse Talk can use to provide more features.

Most importantly, Python is free, well-supported, and a joy to use.

1.3 Free software

Morse Talk is free software; you can redistribute it and/or modify it under the terms of the `MIT Licence`. You are most welcomed to contribute to it over the developing zone https://github.com/orkohunter/morse-talk

1.4 History

Morse Talk was created by *Himanshu Mishra <https://github.com/orkohunter>* in May 2015 as a fun project to work on.

CHAPTER 2

Download

2.1 Software

Source and binary releases: https://pypi.python.org/pypi/morse-talk/

Github (latest development): https://github.com/orkohunter/morse-talk

CHAPTER 3

Installing

Before installing Morse Talk, you need to have setuptools installed.

3.1 Quick install

Get Morse Talk from the Python Package Index at http://pypi.python.org/pypi/morse-talk

or install it with pip

```
pip install morse-talk
```

You can install the development version (at github.com) with

```
git clone https://github.com/orkohunter/morse-talk
cd morse-talk/
python setup.py install
```

3.2 Requirements

3.2.1 Python

To use Morse Talk you need Python 2.7, 3.2 or later.

3.3 Optional packages

Currently Morse Talk does not use any other optional package.

CHAPTER 4

Tutorial

Start here to begin working with Morse Talk.

4.1 Encode a message

Create a message you want to encode.

```
>>> import morse_talk as mtalk
>>> message = "Hi, I'll be there by 10 PM"
```

`message` is a Python string. No leading or trailing whitespaces are allowed in it. Even if it is there, it will be truncated off for encoding.

Now encode your message in Morse Code.

```
>>> code = mtalk.encode(message)
>>> code
'....   ..   --..--       ..   .----.   .-..   .-..       -...   .       -   ....   .   .-.   .
```

Note: Morse Talk supports alphabets, numerals and some special characters only. If any unsupported character is present in the `message`, it will be ignored.

```
>>> message = "Congratualtion!"
>>> mtalk.encode(message)
WARNING: Unsupported characters in the string
'-.-.   ---   -.   --.   .-.   .-   -   ..-   .-..   .-   -   ..   ---   -.   ...'
```

You can also encode the message in binary encoding style.

```
>>> mtalk.encode(code, 'binary')
```

4.2 Decode a message

Decode the message which you have encoded in Morse Code.

```
>>> code = '.-   .-..   .--.   ....   .-       ....-   .....       -..   .   .--.   .-   .-.   -   .
>>> message = mtalk.decode(code)
>>> message
'alpha 45 departed'
```

Decoding a binary message is brainstorming right now. If we think further, there can be multiple messages for a single code. So currently, Morse Talk does not support binary decoding.

```
>>> code = '1110001111111110001111100011111000111111111'
>>> mtalk.decode(code, 'binary')
Sorry, but it seems that binary encodings can have multiple messages. So for now, we couldn't show ev
```

By the way, that code was generated by `mtalk.encode('sorry', 'binary')` ;-)

What Next

Now that you have an idea of what the Morse Talk package provides, you should investigate the parts of the package most useful for you.

CHAPTER 5

Testing

5.1 Requirements for testing

Morse Talk uses the Python nose testing package. If you don't already have that package installed, follow the directions here http://somethingaboutorange.com/mrl/projects/nose

5.2 Testing a source distribution

You can test the complete package from the unpacked source directory with:

```
python setup_egg.py nosetests
```

5.3 Testing an installed package

If you have a file-based (not a Python egg) installation you can test the installed package with

```
>>> import morse_talk
>>> morse_talk.test()
```

or:

```
python -c "import morse_talk; morse_talk.test()"
```

5.4 Testing for developers

You can test any or all of Morse Talk by using the "nosetests" test runner.

First make sure the Morse Talk version you want to test is in your PYTHONPATH (either installed or pointing to your unpacked source directory).

Then you can run individual test files with:

```
nosetests path/to/file
```

or all tests found in dir and an directories contained in dir:

```
nosetests path/to/dir
```

CHAPTER 6

Indices and tables

- genindex
- modindex
- search