yt2mp3 Documentation

Brett Stevenson

Nov 18, 2018

Contents

A program that simplifies the process of searching, downloading and converting Youtube videos to MP3 files from the command-line.

All you need is the video URL or the name of the artist/track you're looking for. The program will attempt to retrieve data for a song matching the provided input by querying the iTunes API and use the data to find a corresponding YouTube video, if a URL is not provided. The video will then be downloaded, converted, and the gathered data will be used to populate the metadata of the MP3.

Note: If a URL is provided and no match is found for the song data, the program will prompt the user for the title/artist and the YouTube thumbnail will be used as the album artwork.

CHAPTER 1

Contributing

If you'd like to contribute to the project, feel free to suggest a feature request and/or submit a pull request.

CHAPTER 2

Contents

2.1 Getting Started

2.1.1 Install

You can install the program with the following command:

```
$ pip install yt2mp3
```

Prerequisites

The program only requires that you have Python 3.4+ and ffmpeg or libav installed. More installation information is available on the additional setup page.

2.1.2 Usage

Once installed, the program can be executed via the command-line as follows:

```
$ yt2mp3 [-options]
```

When the program is finished, you can find the resulting MP3 file in your *Downloads* directory, with the following file-structure `Music/{artist}/{track}.mp3`.

Options

`-t, --track`	Specify the track name query
`-a, --artist`	Specify the artist name query
`-c, --collection`	Specify the album name query
`-u, --url`	Specify a Youtube URL or ID
`-u, --url`	Specify a Youtube URL or ID
`-p, --playlist`	Specify a Youtube playlist URL or ID
`-o, --overwrite`	Overwrite the file if one exists in output directory
`-r, --resolution`	Specify the resolution for the cover-art
`-q, --quiet`	Suppress program command-line output
`-v, --verbose`	Display a command-line progress bar
`--version`	Show the version number and exit
`-h, --help`	Display information on usage and functionality

2.1.3 Contributing

If you'd like to contribute to the project, you can download and install the program with the following commands:

```
# Clone the repository
$ git clone https://github.com/tterb/yt2mp3

# Navigate to the directory
$ cd yt2mp3

# Install program dependencies
$ pip install -r requirements.txt
```

2.2 Additional Setup

The project utilizes the pydub package for audio-file manipulation, which requires that a multimedia decoding library to function properly. You can use ffmpeg or libav.

2.2.1 Mac (using homebrew)

```
# ffmpeg
brew install ffmpeg --with-libvorbis --with-sdl2 --with-theora

####    OR    #####

# libav
brew install libav --with-libvorbis --with-sdl --with-theora
```

2.2.2 Linux (using aptitude)

```
# libav
apt-get install libav-tools libavcodec-extra-53

####    OR    #####

# ffmpeg
apt-get install ffmpeg libavcodec-extra-53
```

2.2.3 Windows

- Download and extract **libav** from Windows binaries provided here.
- Add the **libav** */bin* folder to your PATH

2.3 Changelog

2.3.1 Version 1.2.3

- Improved package CLI
- Modified `requests` version to avoid a potential security vulnerability
- Fixed #22 - Python 3.4 compatibility
- Modified package to use `youtube-dl` for downloading videos
- Enhancement #17 - Added option to specify album name

2.3.2 Version 1.2.0

- Improved package file-structure
- Enhancement #16 - Added custom cover-art resolution option
- Improved default cover-art resolution
- Added validation check for YouTube URLs
- Improved command-line option parsing
- Improved package test coverage and logging
- Improved song data acquisition processes
- Integrated Codecov code coverage
- Improved playlist overwrite handling
- Improve temporary file clean-up on *SystemExit* and *KeyboardInterrupt*
- Removed unnecessary code/comments

2.3.3 Version 1.1.0

- Added support for playlists
- Improved command-line option functionality
- Added duplicate check and overwrite command-line option
- Added Travis CI integration testing
- Improved cover-art resolution
- Improved compatibility for Python < 3.6 versions
- Improved ID3 tag embedding processes

2.3.4 Version 1.0.5

- Added additional error-handling
- Improved song data acquisition for YouTube URLs
- Improved command-line option functionality
- Added additional package documentation
- Fixed issue with missing executable from PyPi

2.3.5 Version 1.0.2

- Published package to PyPi

2.3.6 Version 1.0.0

- All features added
- All bugs created

2.4 Contributing

The following is a set of guidelines for contributing to this project. These are not rules, use your best judgment and feel free to propose changes to this document in a pull request.

2.4.1 Code of Conduct

This project adheres to the Contributor Covenant code of conduct. By participating, you are expected to uphold this code. Please report unacceptable behavior here.

2.4.2 How Can I Contribute?

Reporting Bugs

This section guides you through submitting a bug report for Atom. Following these guidelines helps maintainers and the community understand your report, reproduce the behavior, and find related reports.

- Before creating bug reports, please check the existing issues as you might find out that one has already been created by another user.
- When you are creating a bug report, please provide a clear and descriptive title.
- Please include as many details as possible in your report. Remember, in order to fix the issue, we must first be able to reproduce it.
- The use of images, GIFs, or any other form of media is not only *allowed*, it's **encouraged**.
- Include any relevant details about your configuration and environment.

Enhancements & Feature Requests

Users provide the best insight for improving a product. So always feel free to suggest improvements, feedback, and give the developers an idea of what new features you'd like to see in the future. **Don't forget, this is one of the greatest benefits of open-source development!** If you'd like to submit an enhancement request, please include the following information: * Current project version (check that you are using the latest version) * How would this enhancement benefit most users? * Do you have any examples of other applications that feature this enhancement? * Relevant details about your configuration and environment. * **Note:** Don't forget to tag your post with the *Enhancement* label.

Pull Requests

Working on your first Pull Request? You can learn how from this *free* series How to Contribute to an Open Source Project on GitHub

- Provide as much information as possible on the changes made to existing code.
- Include thoughtfully-worded, well-structured commit messages and comments.
- Include screenshots and animated GIFs in your pull request whenever possible.
- When addressing an existing issue, please include an issue mention in the description of your pull request.
- End files with a newline.
- Avoid platform-dependent code on cross-platform applications.
- Ensure that submitted code follows the style guidlines specified in the Styleguide below.

2.4.3 Styleguides

Commit Messages

- Do your best to conform the tense to that of the existing commits. ("Add feature" not "Added feature" or vice versa)
- Use the imperative mood ("Move cursor to…" not "Moves cursor to…")
- **Consider starting the commit message with an applicable emoji:**
  - `:art:` when improving the format/structure of the code
  - `:racehorse:` when improving performance
  - `:memo:` when writing docs
  - `:penguin:` when fixing something on Linux
  - `:apple:` when fixing something on macOS
  - `:checkered_flag:` when fixing something on Windows
  - `:bug:` when fixing a bug
  - `:recycle:` when refactoring code
  - `:fire:` when removing code or files
  - `:white_check_mark:` when adding tests
  - `:lock:` when dealing with security
  - `:iphone:` when working on responsive design
  - `:truck` when moving or renaming files
  - `:bento:` when adding or updating assets
  - `:building_construction:` when making architectural changes
  - `:wheelchair:` when improving accessibility
  - `:construction_worker:` when modifying the CI build
  - `:non-potable_water:` when plugging memory leaks
  - `:arrow_up:` when upgrading dependencies
  - `:arrow_down:` when downgrading dependencies
  - `:rotating_light:` when removing linter warnings

Code

- **Use spaces around operators**
  - `count + 1` instead of `count+1`
- **Capitalize initialisms and acronyms in names, except for the first word, which should be lower-case:**
  - `getURL` instead of `getUrl`
- Provide comments for important or complex code
- Conform your code to the existing styles to the best of your ability

- Always check that your submission follows The Boy Scout Rule

2.5 The MIT License (MIT)

License taken from MIT license.

CHAPTER 3

Indices and tables

- genindex
- search