

---

# **enpassreaderlib Documentation**

***Release 0.2.2***

**Costas Tyfoxylos**

**Mar 02, 2023**



# CONTENTS

<b>1</b>	<b>enpassreaderlib</b>	<b>3</b>
1.1	Development Workflow . . . . .	3
1.2	Important Information . . . . .	4
1.3	Project Features . . . . .	4
<b>2</b>	<b>Installation</b>	<b>5</b>
<b>3</b>	<b>Usage</b>	<b>7</b>
<b>4</b>	<b>Contributing</b>	<b>9</b>
4.1	Submit Feedback . . . . .	9
<b>5</b>	<b>enpassreaderlib</b>	<b>11</b>
5.1	enpassreaderlib package . . . . .	11
<b>6</b>	<b>Credits</b>	<b>13</b>
6.1	Development Lead . . . . .	13
6.2	Contributors . . . . .	13
<b>7</b>	<b>History</b>	<b>15</b>
<b>8</b>	<b>0.0.1 (25-03-2021)</b>	<b>17</b>
<b>9</b>	<b>0.1.0 (25-03-2021)</b>	<b>19</b>
<b>10</b>	<b>0.1.1 (25-03-2021)</b>	<b>21</b>
<b>11</b>	<b>0.1.2 (07-07-2021)</b>	<b>23</b>
<b>12</b>	<b>0.2.0 (02-03-2023)</b>	<b>25</b>
<b>13</b>	<b>0.2.1 (02-03-2023)</b>	<b>27</b>
<b>14</b>	<b>0.2.2 (02-03-2023)</b>	<b>29</b>
<b>15</b>	<b>Indices and tables</b>	<b>31</b>
	<b>Python Module Index</b>	<b>33</b>
	<b>Index</b>	<b>35</b>



Contents:



## ENPASSREADERLIB

A library implementing the decrypting and retrieving secrets from an enpass 6 database.

- Documentation: <https://enpassreaderlib.readthedocs.org/en/latest>

### 1.1 Development Workflow

The workflow supports the following steps

- lint
- test
- build
- document
- upload
- graph

These actions are supported out of the box by the corresponding scripts under `_CI/scripts` directory with sane defaults based on best practices. Sourcing `setup_aliases.ps1` for windows powershell or `setup_aliases.sh` in bash on Mac or Linux will provide with handy aliases for the shell of all those commands prepended with an underscore.

The bootstrap script creates a `.venv` directory inside the project directory hosting the virtual environment. It uses `pipenv` for that. It is called by all other scripts before they do anything. So one could simple start by calling `_lint` and that would set up everything before it tried to actually lint the project

Once the code is ready to be delivered the `_tag` script should be called accepting one of three arguments, `patch`, `minor`, `major` following the semantic versioning scheme. So for the initial delivery one would call

```
$ _tag -minor
```

which would bump the version of the project to 0.1.0 tag it in git and do a push and also ask for the change and automatically update `HISTORY.rst` with the version and the change provided.

So the full workflow after git is initialized is:

- repeat as necessary (of course it could be `test - code - lint` :) )
  - code
  - lint
  - test
- commit and push
- develop more through the code-lint-test cycle

- tag (with the appropriate argument)
- build
- upload (if you want to host your package in pypi)
- document (of course this could be run at any point)

## 1.2 Important Information

This template is based on pipenv. In order to be compatible with requirements.txt so the actual created package can be used by any part of the existing python ecosystem some hacks were needed. So when building a package out of this **do not** simple call

```
$ python setup.py sdist bdist_egg
```

**as this will produce an unusable artifact with files missing.** Instead use the provided build and upload scripts that create all the necessary files in the artifact.

## 1.3 Project Features

See USAGE.rst.

- Can retrieve single entries
- Can iterate over all entries
- Can do fuzzy matching of entries while searching



## INSTALLATION

At the command line:

```
$ pip install enpassreaderlib
```

Or, if you have virtualenvwrapper installed:

```
$ mkvirtualenv enpassreaderlib  
$ pip install enpassreaderlib
```

Or, if you are using pipenv:

```
$ pipenv install enpassreaderlib
```

Or, if you are using pipx:

```
$ pipx install enpassreaderlib
```

Important note for pysqlcipher3:

pysqlcipher3 needs to compile on your workstation and it might not succeed if header files are missing. On my Mac I had to follow something like the below process:

```
brew install sqlcipher  
# Assuming the version installed is 4.4.3 adjust accordingly  
export C_INCLUDE_PATH=$BREWPATH/Cellar/sqlcipher/4.4.3/include  
export LIBRARY_PATH=$BREWPATH/Cellar/sqlcipher/4.4.3/lib  
# Activate the virtual environment that the project is installed to fix the installation  
. .venv/bin/activate  
pip install pysqlcipher3
```

On linux based systems probably sqlcipher-dev will need to be installed for the package to successfully compile.



**USAGE**

To develop on enpassreaderlib:

```
# The following commands require pipenv as a dependency

# To lint the project
_CI/scripts/lint.py

# To execute the testing
_CI/scripts/test.py

# To create a graph of the package and dependency tree
_CI/scripts/graph.py

# To build a package of the project under the directory "dist/"
_CI/scripts/build.py

# To see the package version
_CI/scripts/tag.py

# To bump semantic versioning [--major|--minor|--patch]
_CI/scripts/tag.py --major|--minor|--patch

# To upload the project to a pypi repo if user and password are properly provided
_CI/scripts/upload.py

# To build the documentation of the project
_CI/scripts/document.py
```

To use enpassreaderlib in a project:

```
from enpassreaderlib import EnpassDB
enpass = EnpassDB('db_file_path', 'db_master_password', 'optional_key_file')

# Get a specific entry
entry = enpass.get_entry('ENTRY_TITLE')
entry.password

# Search with fuzzy searching
for entry in enpass.search_entries('SOME_PART_OF_A_PASSWORD_TITLE'):
    print(f'{entry.title} {entry.password}')
```

(continues on next page)

(continued from previous page)

```
# Iterate over all the entries of the database
for entry in enpass.entries:
    print(f'{entry.title} {entry.password}')
```

## CONTRIBUTING

# Based on work shown at <https://github.com/HazCod/enpass-cli/issues/16>

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

### 4.1 Submit Feedback

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.

#### 4.1.1 Get Started!

Ready to contribute? Here's how to set up *enpassreaderlib* for local development. Using of *pipenv* is highly recommended.

1. Clone your fork locally:

```
$ git clone https://github.com/costastf/enpassreaderlib
```

2. Install your local copy into a virtualenv. Assuming you have *pipenv* installed, this is how you set up your clone for local development:

```
$ cd enpassreaderlib/  
$ pipenv install --ignore-pipfile
```

3. Create a branch for local development:

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

Now you can make your changes locally. Do your development while using the CI capabilities and making sure the code passes lint, test, build and document stages.

4. Commit your changes and push your branch to the server:

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

5. Submit a merge request



## ENPASSREADERLIB

### 5.1 enpassreaderlib package

#### 5.1.1 Submodules

#### 5.1.2 enpassreaderlib.enpassreaderlib module

Main code for enpassreaderlib.

```
class enpassreaderlib.enpassreaderlib.EnpassDB(database_path, password, keyfile=None,  
                                              pbkdf2_rounds=100000)
```

Bases: object

Manages the database object exposing useful methods to interact with it.

**property cipher\_key**

The cipher key to decrypt entries in the database.

**Returns**

The cipher key to decrypt the database entries.

**Return type**

cipher\_key (string)

**property entries**

All the entries in the database.

**Returns**

The password entries in the database.

**Return type**

entries (list)

**get\_entry**(*name*)

Retrieves a single entry matching the name.

**Parameters**

**name** – The name of the password entry to retrieve.

**Returns**

A password entry object if match found else None.

**Return type**

entry (*Entry*)

**property master\_password**

The master password calculated along with the key if provided else the password provided.

**Returns**

The master password to decrypt the database.

**Return type**

master\_password (bytearray)

**search\_entries(name)**

Retrieves any entry that matches the name provided (fuzzy matching).

**Parameters**

**name** – The name to search the password entries for.

**Returns**

A list of password entries matching the fuzzy search for the given name.

**Return type**

entries (list)

**class enpassreaderlib.enpassreaderlib.Entry(database\_row)**

Bases: object

Models a password entry and exposes some useful attributes about it.

**property password**

The plaintext password of the entry.

**Returns**

The plaintext password of the entry.

**Return type**

password (text)

**property totp\_seed**

### 5.1.3 enpassreaderlib.enpassreaderlibexceptions module

Custom exception code for enpassreaderlib.

**exception enpassreaderlib.enpassreaderlibexceptions.EnpassDatabaseError**

Bases: Exception

The database is not a valid enpass 6 database or master password or key provided cannot decrypt it.

### 5.1.4 Module contents

enpassreaderlib package.

Import all parts from enpassreaderlib here



CREDITS

## 6.1 Development Lead

- Costas Tyfoxylos <[costas.tyf@gmail.com](mailto:costas.tyf@gmail.com)>

## 6.2 Contributors

None yet. Why not be the first?



---

CHAPTER  
**SEVEN**

---

**HISTORY**



**0.0.1 (25-03-2021)**

- First code creation



**0.1.0 (25-03-2021)**

- First release with basic required functionality.





### 0.1.1 (25-03-2021)

- Loosely pinned dependencies and updated the usage and installation notes a bit.



**0.1.2 (07-07-2021)**

- Added pipeline.



**0.2.0 (02-03-2023)**

- Expose totp seeds for passwords that support it.



**0.2.1 (02-03-2023)**

- Fix linting.





**0.2.2 (02-03-2023)**

- Fix for entries with no password.



## INDICES AND TABLES

- `genindex`
- `modindex`
- `search`



## PYTHON MODULE INDEX

### e

`enpassreaderlib`, [12](#)

`enpassreaderlib.enpassreaderlib`, [11](#)

`enpassreaderlib.enpassreaderlibexceptions`, [12](#)



## INDEX

### C

`cipher_key` (*enpassreaderlib.EnpassDB* property), 11

### E

`EnpassDatabaseError`, 12

`EnpassDB` (*class in enpassreaderlib.enpassreaderlib*), 11

`enpassreaderlib`

module, 12

`enpassreaderlib.enpassreaderlib`

module, 11

`enpassreaderlib.enpassreaderlibexceptions`

module, 12

`entries` (*enpassreaderlib.enpassreaderlib.EnpassDB* property), 11

`Entry` (*class in enpassreaderlib.enpassreaderlib*), 12

### G

`get_entry()` (*enpassreaderlib.enpassreaderlib.EnpassDB* method), 11

### M

`master_password` (*enpassreaderlib.enpassreaderlib.EnpassDB* property), 11

module

`enpassreaderlib`, 12

`enpassreaderlib.enpassreaderlib`, 11

`enpassreaderlib.enpassreaderlibexceptions`, 12

### P

`password` (*enpassreaderlib.enpassreaderlib.Entry* property), 12

### S

`search_entries()` (*enpassreaderlib.enpassreaderlib.EnpassDB* method), 12

### T

`totp_seed` (*enpassreaderlib.enpassreaderlib.Entry* property), 12