pwman3 Documentation

Release 0.11.0

Oz Tiram

May 05, 2020

Contents

1 Installing and upgrading: 3
- 1.1 Upgrading from version 0.5.x . . . 3
- 1.2 Upgrading from version 0.6 or later: . . . 4
- 1.3 A Note about Python versions . . . 4
- 1.4 Database versions . . . 4

2 Pwman3 - tutorial 5

3 Configuring Pwman3 9

4 Packaging for UNIX\Linux 11
- 4.1 Requirements and suggested packages . . . 11
- 4.2 Building a man page . . . 11

5 pwman Package 13
- 5.1 `pwman` Package . . . 13
- 5.2 Subpackages . . . 13

6 About pwman3: 23

Python Module Index 25

Index 27

Contents:

CHAPTER 1

Installing and upgrading:

You can get the latest stable version of pwman3 from pypi:

```
$ pip install pwman3
```

Or you can install pwman3 cloning the repository and running:

```
$ git clone https://github.com/pwman3/pwman3.git
$ cd pwman3 && pip install .
```

Inside the extraced directory. At the moment pypi still does not have the latest version of pwman3.

1.1 Upgrading from version 0.5.x

If you used the 0.5.x version of `pwman3` your database is not compatible with the new 0.6.x and later versions of `pwman3`. You need to export your database to a CSV from version 0.5.3 with:

```
pwman> export
```

See `help export` when running pwman3 in version 0.5.3. Once exported you should rename your old database, to keep a backup of it. Then you can install pwman3 in version 0.6.x or later as described above. When finished you can import your passwords from the CSV to a new database with:

```
$ pwman3 -i path_to_your_file.csv \;
```

The *;* tells the importer that the file is semi-colon separated. When the import is done, start pwman3 with:

```
$ pwman3
```

If the import was success, erase the CSV file, which contains your passwords in clear text.

1.2 Upgrading from version 0.6 or later:

Once again the latest release (version 0.9) breaks compatibility with earlier versions. As a user you might consider this an annoyance, which is understandable. However, older Pwman3 version used a very old cryptography library which is no longer actively maintained. With the migration of the code base to use the cryptography library, there was a necessary change of the underlying encryption algorithm. AES encryption with ECB mode was the old algorithm used, which is by now considered outdated, and it was replaced with Fernet encryption which is considered best practice.

To upgrade, follow the path described above to export your database to a CSV, and the import it.

1.3 A Note about Python versions

Earlier versions of Pwman3 supported both Python2 and Python3 versions. This created a bit of an effort for maintaining version compatablity, and served as a migration path for future versions. Python 3 has been around now for quite a while, and soon enough, Python2 support will end. Python 3 is mature enough, and offers many improvements for developers. Python 3.4 is included in all major modern Linux distributions. If you use a certain enterprise Linux versions which does not ship Python 3.3 or later, the process for installing a newer Python versions is pretty straight forward and very well documented. You should opt for using newer Python versions for all your software if possible.

If you want to learn more about why Python 3, see the following Python3 statement

1.4 Database versions

The current version of Pwman3 is tested with Postgresql-9.3, MySQL-5.5, MongoDB 2.6.X and SQLite3.

The required python drivers are:

- pymysql version 0.6.6
- psycopg2 version 2.6
- pymongo version 2.8

CHAPTER 2

Pwman3 - tutorial

Pwman3 is a text centric password manager, and as such suitable for people wishing to use it over SSH or brail terminals.

Pwman3 is launched by typing `pwman3` in your shell. Multiple start options are available. You can see them by typing:

```
$ pwman3 -h
```

For more information see also `man pwman3`.

When started for the first time, `pwman3` will create a configuration file and your database (By default an SQLite database called `pwman.db`) in your home directory under `.pwman`. Before creating the database you will be asked to enter the master password which will be used to create an encryption key which will be used to encrypt the entries in your database:

```
$ pwman3
Please type in the master password:
```

Take note of this password! If you forget it, you won't be able to read your password database.

Now `pwman3` will wait for your input. The user interface is a simple loop waiting for user commands. You can see all the commands by typing:

```
pwman> help

Documented commands (type help <topic>):
========================================
cls   delete  exit    forget  list  open    print
copy  edit    export  help    new   passwd  tags

Aliases:
========
EOF  cp  h  ls  n  o  p  rm
```

Most commands have a single or two letter alias which is easy to remember.

As for a start you would probably like to store some passwords in your database, all you need to do is type the command `new` (or the alias `n`), and then insert the information as prompted:

```
pwman> new
Username: oz123
Password:
Url: example.org
Notes: Your notes
Tags: bank, private, my tag
```

Note the password is typed in without echoing it. Also note that two tags were separated by a space. Now, you can list your database entries with:

> pwman> ls
>
> ID USER URL TAGS 1 oz123 example.org bank, private, my tag

After a while you have had many new nodes inserted in your database:

```
ID  USER        URL                     TAGS
1   oz123       example.org             bank, private, my tag
2   oz123       intranet.workplace.biz  work
3   oz123       shopping.com            shopping
```

Now searching for a searching entry can become quite tiresome. Hence, you can minimize the number of entries display by applying a filter to the `list` command:

```
pwman> ls work

ID  USER        URL                 TAGS
2   oz123       intranet.workplace.biz  workplace
```

If you don't remember all the tags you created you can list the tags using the command `tags`:

```
pwman> tags
Tags:
bank,
private,
shopping
```

To see the content of a saved entry, you need to use the `print` command:

```
pwman> print 2
Username: oz123
Password: Org4n7#
URL: intranet.workplace.biz
Notes:
Tags: 'work'
Type Enter to flush screen or wait 5 sec.
```

You can also search an entry by typing part of the URL:

```
pwman> ls u:twit
ID  USER                    URL                     TAGS
115 oz123@mymail.com        twitter.com             web
```

Notice, that the `print` command expects an entry number as an argument. After printing the content of the entry, pwman3 will wait for `Enter` to be pressed or 5 seconds until it flushes the screen. This way, unauthorized eyes can not browse your screen and see your password. You can always scroll up to see the printed entry or print it again.

If you don't want to type passwords and URLs constantly `Pwman3` comes with two shortcut commands. The first shortcut is `open`, when calling it with an entry number it will open the URL in your default browser:

```
pwman> open 2
```

This will open the URL *intranet.workplace.biz*. After opening the browser you can use the `copy` command to copy your password to your clipboard. The password will be *erased* from the clipboard after 10 seconds:

```
pwman> copy 2
erasing in 10 sec...
```

After working with passwords for quite a while you can `delete` (or `rm`) entries or edit them:

```
pwman> rm 2
Are you sure you want to delete node 2 [y/N]?N

pwman> e 2
Editing node 2.
1 - Username: oz123
2 - Password: Org4n7#
3 - Url: intranet.workplace.biz
4 - Notes:
5 - Tags: workplace
X - Finish editing
Enter your choice:
```

You now know all the basics of using `pwman3`. If you need more help, try the command `help` to see more commands which are not documented here. Alternatively, you can open a ticket in https://github.com/pwman3/pwman3/issues.

CHAPTER 3

Configuring Pwman3

By default Pwman3 will read the configuration file from the following path on Unix like systems:

```
~/.config/pwman/config
```

On windows the configuration is found in `%APPDATA/pwman/%`

You can override this by giving the `-c <Config File>` at the commnad line when starting Pwman3.

The configuration file has the following structure:

```
[Section]
Option = Value
```

The following is an example default config file:

```
[Readline]
history = <PWMAN_DATA_DIR>/history

[Global]
save = True
colors = yes
cp_timeout = 5
umask = 0100
cls_timeout = 10
xsel = /usr/bin/xsel
lock_timeout = 60

[Database]
filename = sqlite:///<PWMAN_DATA_DIR>/pwman.db`

[Updater]
supress_version_check = no
client_info = ee5cd64310568736b971e3fb7c7064a4459b99a2b78672515fd0f06c82f65d5
```

On Windows `PWMAN_DATA_DIR` is `%APPDATA/pwman/%` on Unix like systems it is

`~/.local/share/pwman/`.

A detailed table of all configuration parameters is found in *pwman.util.config.Config*.

CHAPTER 4

Packaging for UNIX\Linux

4.1 Requirements and suggested packages

The most basic install of pwman requires Python 2.7.x or Python 3.x with SQLite support. Some users might be interested in working with a network database such as MySQL or Postgreql, in this case the package should suggest to install but not depend on the following components:

- python-psycopg2
- python-pymysql

4.2 Building a man page

Many users of command line programs expect to find a man page for each and every program installed. Some Linux flavours has even made it a policy (for example Debian Policy).

If you would like to format a man page for Pwman3, this program provides a helper method for that:

```
$ python setup.py build_manpage
```

This will create a man page *pwman3.1* in the directory where you issued the command. You can control the output by editing the option *output* in the file *setup.cfg*.

CHAPTER 5

pwman Package

5.1 pwman Package

class pwman.__init__.**PkgMetadata**
Bases: object

pwman.__init__.**calculate_client_info**()

pwman.__init__.**get_conf**(*args*)

pwman.__init__.**get_conf_options**(*args*, *OSX*)

pwman.__init__.**get_db_version**(*config*, *args*)

pwman.__init__.**is_latest_version**(*version*, *client_info*)
check current version againt latest version

pwman.__init__.**parser_options**(*formatter_class=<class 'argparse.HelpFormatter'>*)

pwman.__init__.**set_db**(*args*, *configp*)

pwman.__init__.**set_umask**(*configp*)

pwman.__init__.**set_xsel**(*configp*, *OSX*)

5.2 Subpackages

5.2.1 data Package

convertdb Module

class pwman.data.convertdb.**DBConverter**(*args*, *config*)
Bases: object

A general class to provide a base template for converting a database from one version to another

backup_old_db()

convert_nodes()

create_new_db(*new_version=0.4*)

static detect_db_version(*filename*)
This method should accept a pwman db file name, and it should try to detect which database version it is.

static invoke_converter(*dbversion*, *future_version*)
this method should accept the two parameters and according to them invoke the right converter

print_success()

read_old_db()

run()

save_new_nodes_to_db()

save_old_key()

class pwman.data.convertdb.**PwmanConvertKey**(*args*, *config*)
Bases: *pwman.data.convertdb.DBConverter*

read_old_db()

save_old_key()

database Module

class pwman.data.database.**Database**
Bases: object

add_node(*node*)

close()

editnode(*nid*, ***kwargs*)

fetch_crypto_info()

get_node(*id*)

get_user_password()
get the databases password from the user

getnodes(*ids*)

lazy_get_nodes(*ids*)
iterates thought ids and yield a node for each id

lazy_list_node_ids(*filter=None*)
return a generator that yields the node ids

listtags()

loadkey()
return _keycrypted

open(*dbver=None*)
Open the database, by calling the _open method of the class inherited for the specific database. When done validation that the file is OK, check if it has encryption key, by calling enc = CryptoEngine.get() key = self.loadkey()

removenodes(*nid*)

save_crypto_info(*seed*, *digest*)
save the random seed and the digested key

savekey(*key*)

exception pwman.data.database.**DatabaseException**
Bases: Exception

factory Module

pwman.data.factory.**check_db_version**(*dburi*)

pwman.data.factory.**createdb**(*dburi*, *version*)

pwman.data.factory.**no_parse_uri**(*dburi*)

pwman.data.factory.**parse_postgres_uri**(*dburi*)

pwman.data.factory.**parse_sqlite_uri**(*dburi*)
return dburi.netloc if on windows, because this was someone break

nodes Module

class pwman.data.nodes.**Node**(*clear_text=True*, ***kwargs*)
Bases: object

classmethod from_encrypted_entries(*username*, *password*, *url*, *notes*, *tags*)
We use this alternatively, to create a node instance when reading the encrypted entities from the database

notes
Get the current notes.

password
Get the current password.

tags

to_encdict()
Return a dictionary of encrypted records

url
Get the current url.

username
Get the current username.

Subpackages

drivers Package

sqlite Module

SQLite Database implementation.

class pwman.data.drivers.sqlite.**SQLite**(*filename*, *dbformat=0.6*)
Bases: *pwman.data.database.Database*

classmethod check_db_version(*fname*)
check the database version.

5.2.2 exchange Package

importer Module

A module to hold the importer class

class pwman.exchange.importer.**BaseImporter**

Bases: object

The base class which defines the action needed to import data to pwman database

class pwman.exchange.importer.**CSVImporter**(*args*, *config*, *db*)

Bases: *pwman.exchange.importer.BaseImporter*

A reference implementation which imports a CSV to the pwman database

run(*callback=<class 'pwman.ui.tools.CLICallback'>*)

class pwman.exchange.importer.**Importer**(*args*, *invoke=<class 'pwman.exchange.importer.CSVImporter'>*)

Bases: object

The actual job runner which by default runs a csv importer instance. This could be changes by calling other instance which for example import from KeePass XML or other formats.

run()

5.2.3 ui Package

ui Package

baseui Module

class pwman.ui.baseui.**AliasesMixin**

Define all the alias you want here...

do_EOF(*args*)

do_cp(*args*)

do_e(*args*)

do_h(*arg*)

do_l(*args*)

do_ls(*args*)

do_n(*arg*)

do_o(*args*)

do_p(*arg*)

do_rm(*arg*)

class pwman.ui.baseui.**BaseCommands**

do_EOF(*args*)

do_cls(*args*)

clear the screen

do_copy(*args*)
copy item to clipboard. Accepts an id or *u:<URL substring>*

do_delete(*args*)

do_edit(*args*, *menu=None*)

do_exit(*args*)
close the text console

do_export(*args*)
export the database to a given format

do_forget(*args*)
drop saved key forcing the user to re-enter the master password

do_info(*args*)

do_list(*args*)
list all existing nodes in database

do_new(*args*)

do_open(*args*)

do_passwd(*args*)
change the master password of the database

do_pp(*args*)
print the password only

do_print(*args*)

do_tags(*args*)
print all existing tags

error(*exception*)

class pwman.ui.baseui.**BaseUtilsMixin**
Helper class for storing all privates, useful for testing these methods directly

class pwman.ui.baseui.**HelpUIMixin**
this class holds all the UI help functionality. in PwmanCliNew. The later inherits from this class and allows it to print help messages to the console.

help_cls()

help_copy()

help_delete()

help_edit()

help_exit()

help_export()

help_forget()

help_help()

help_info()

help_list()

help_new()

help_open()

help_passwd()

help_print()

help_tags()

cli Module

class pwman.ui.cli.**PwmanCli**(*db*, *hasxsel*, *callback*, *config_parser*, ***kwargs*)
Inherit from the BaseCommands and Aliases

undoc_header = 'Aliases:'

pwman.ui.cli.**check_version**(*version*, *client_info*)

pwman.ui.cli.**get_ui_platform**(*platform*)

pwman.ui.cli.**main**()

mac Module

class pwman.ui.mac.**PwmanCliMac**(*db*, *hasxsel*, *callback*, *config_parser*, ***kwargs*)
inherit from PwmanCli, override the right functions...

do_copy(*args*)
copy item to clipboard. Accepts an id or *u:<URL substring>*

tools Module

Define the CLI interface for pwman3 and the helper functions

class pwman.ui.tools.**ANSI**
ANSI Colors

Black = 30

Blue = 34

Bold = 1

Cyan = 36

Green = 32

Magenta = 35

Red = 31

Reset = 0

Underscore = 2

White = 37

Yellow = 33

class pwman.ui.tools.**CLICallback**

getinput(*question*)
Return text

getnewsecret(*question*)

getsecret(*question*)
Return key

class pwman.ui.tools.**CMDLoop**(*config*)
The menu that drives editing of a node

add(*item*)

run(*new_node=None*, *reader=<built-in function input>*)

class pwman.ui.tools.**CliMenuItem**(*name*, *getter*)

pwman.ui.tools.**get_or_create_pass**()

pwman.ui.tools.**getinput**(*question*, *default=''*, *reader=<built-in function input>*, *completer=None*, *width=10*, *drop='drop'*)
http://stackoverflow.com/questions/2617057/ supply-inputs-to-python-unittests

pwman.ui.tools.**open_url**(*link*, *macosx=False*)
launch xdg-open or open in MacOSX with url

pwman.ui.tools.**set_selection**(*new_node*, *items*, *selection*, *reader*)

pwman.ui.tools.**text_to_clipboards**(*text*)
copy text to clipboard credit: https://pythonadventures.wordpress.com/tag/xclip/

pwman.ui.tools.**text_to_mcclipboard**(*text*)
copy text to mac os x clip board credit: https://pythonadventures.wordpress.com/tag/xclip/

pwman.ui.tools.**typeset**(*text*, *color*, *bold=False*, *underline=False*, *has_colorama=True*)
print colored strings using colorama

win Module

Windows UI components

class pwman.ui.win.**PwmanCliWin**(*db*, *hasxsel*, *callback*, *config_parser*, ***kwargs*)
windows ui class

do_cls(*args*)
clear the screen

do_copy(*args*)
copy item to clipboard. Accepts an id or *u:<URL substring>*

do_print(*args*)

pwman.ui.win.**heardEnterWin**()

pwman.ui.win.**winGetClipboard**()

pwman.ui.win.**winSetClipboard**(*text*)

5.2.4 util Package

callback Module

Callback interface To be used when UI needs to be back to get info from user.

class pwman.util.callback.**CLICallback**

getinput(*question*)
Return text

getnewsecret(*question*)

getsecret(*question*)
Return key

class pwman.util.callback.**Callback**
Callback interface. Callback classes must implement this.

error(*error*)
Present error to user

getinput(*question*)
Return text

getsecret(*question*)
Return key

notice(*warning*)
Present notice to user

warning(*warning*)
Present warning to user

config Module

class pwman.util.config.**Config**(*filename=None*, *defaults=None*, ***kwargs*)
The configuration of pwman is done with an *ini* file found in XDG_CONFIG_HOME on Unix systems. On windows the configuration is found in `%APPDATA/pwman/%` The following describe the possible sections in the file and the default values of each parameter:

Section	*Readline*
history	path to the file containing history of commands typed
Section	*Global*
save	True or False - whether the Configuring file should be saved
colors	yes or no - If set to *no*, no colors used in output. This is useful for breil terminals.
cp_timeout	Number of seconds before the clipboard is erased.
cls_timeout	Number of seconds before the screen is clean after a print. Set to 0 to disable.
umask	The umask in which database and configuration files are written.
xsel	path to the xsel binary (LinuxBSD only).
lock_timeout	sets the period (in secods) after which the database is locked.
Section	*Database*
dburi	Database URI conforming to *RFC3986*. SQLite, Postgreql, MySQL and MongoDB are currently supported. **SQLite example:** *sqlite:///path/to/your/db* **Postgreql example:** *postgresql://<user>:<pass>@<host[:port]>/<database>* **MySQL example:** *mysql://<user>:<pass>@<host[:port]>/<database>* **MongoDB example:** *mongodb://<user>:<pass>@<host[:port]>/<database>*
Section	*Updater*
supress_version_check	yes or no - check for newer versions of pwman3
client_info	sha256 digest of host name and username, used for identifying the client
Section	*UI*
URL_length	**22 - the max length of URL to show.** Longer URLs are trimmed
URL_pad	25 - the padding of the URL in *line_format*
user_pad	**25 - the padding of the user name in** *line_format*
tag_pad	20 - the padding of tags in the *line_format*
line_format	*{ID:<3} {USER:<{user_pad}}{URL:<{url_pad}}{Tags:<{tag_pad}}* # noqa

get_value(*section*, *name*)

save()

set_value(*section*, *name*, *value*)

exception pwman.util.config.**ConfigException**(*message*)

Basic exception for config.

exception pwman.util.config.**ConfigNoConfigException**(*message*)

`pwman.util.config.find_config_dir`
Backward compatibly config dir finder

If ~/.appname is not found define a new XDG compat one

`pwman.util.config.get_pass_conf(config)`

crypto_engine Module

class `pwman.util.crypto_engine.CryptoEngine(salt=None, digest=None, algorithm='AES', timeout=-1, reader=None)`

authenticate(*password*)
salt and digest are stored in a file or a database

callback
return call back function

changepassword(*reader=<built-in function input>*)

decrypt(*cipher_text*)

encrypt(*text*)

forget()
discard cipher

classmethod get(*timeout=-1*)

get_salt_digest()

set_salt_digest(*key*)

exception `pwman.util.crypto_engine.CryptoException`

`pwman.util.crypto_engine.decode_AES(cipher, encoded_text)`

`pwman.util.crypto_engine.encode_AES(cipher, clear_text)`

`pwman.util.crypto_engine.generate_password(pass_len=8, uppercase=True, lowercase=True, digits=True, special_chars=True)`

`pwman.util.crypto_engine.get_cipher(password, salt)`
Create a chiper object from a hashed password

`pwman.util.crypto_engine.get_digest(password, salt)`
Get a digest based on clear text password

`pwman.util.crypto_engine.prepare_data(text, block_size)`
prepare data before encryption so the lenght matches the expected lenght by the algorithm.

`pwman.util.crypto_engine.zerome(string)`
securely erase strings … for windows: ctypes.cdll.msvcrt.memset

CHAPTER 6

About pwman3:

Pwman3 is a command line password manager for Python3 with support for multiple databases.

Python Module Index

p

pwman.__init__, 13
pwman.data.convertdb, 13
pwman.data.database, 14
pwman.data.drivers.sqlite, 15
pwman.data.factory, 15
pwman.data.nodes, 15
pwman.exchange.importer, 16
pwman.ui, 16
pwman.ui.baseui, 16
pwman.ui.cli, 18
pwman.ui.mac, 18
pwman.ui.tools, 18
pwman.ui.win, 19
pwman.util.callback, 19
pwman.util.config, 20
pwman.util.crypto_engine, 22

Index

A

add() (*pwman.ui.tools.CMDLoop method*), 19
add_node() (*pwman.data.database.Database method*), 14
AliasesMixin (*class in pwman.ui.baseui*), 16
ANSI (*class in pwman.ui.tools*), 18
authenticate() (*pwman.util.crypto_engine.CryptoEngine method*), 22

B

backup_old_db() (*pwman.data.convertdb.DBConverter method*), 13
BaseCommands (*class in pwman.ui.baseui*), 16
BaseImporter (*class in pwman.exchange.importer*), 16
BaseUtilsMixin (*class in pwman.ui.baseui*), 17
Black (*pwman.ui.tools.ANSI attribute*), 18
Blue (*pwman.ui.tools.ANSI attribute*), 18
Bold (*pwman.ui.tools.ANSI attribute*), 18

C

calculate_client_info() (*in module pwman.__init__*), 13
Callback (*class in pwman.util.callback*), 20
callback (*pwman.util.crypto_engine.CryptoEngine attribute*), 22
changepassword() (*pwman.util.crypto_engine.CryptoEngine method*), 22
check_db_version() (*in module pwman.data.factory*), 15
check_db_version() (*pwman.data.drivers.sqlite.SQLite class method*), 15
check_version() (*in module pwman.ui.cli*), 18
CLICallback (*class in pwman.ui.tools*), 18
CLICallback (*class in pwman.util.callback*), 19
CliMenuItem (*class in pwman.ui.tools*), 19
close() (*pwman.data.database.Database method*), 14
CMDLoop (*class in pwman.ui.tools*), 19
Config (*class in pwman.util.config*), 20
ConfigException, 21
ConfigNoConfigException, 21
convert_nodes() (*pwman.data.convertdb.DBConverter method*), 14
create_new_db() (*pwman.data.convertdb.DBConverter method*), 14
createdb() (*in module pwman.data.factory*), 15
CryptoEngine (*class in pwman.util.crypto_engine*), 22
CryptoException, 22
CSVImporter (*class in pwman.exchange.importer*), 16
Cyan (*pwman.ui.tools.ANSI attribute*), 18

D

Database (*class in pwman.data.database*), 14
DatabaseException, 15
DBConverter (*class in pwman.data.convertdb*), 13
decode_AES() (*in module pwman.util.crypto_engine*), 22
decrypt() (*pwman.util.crypto_engine.CryptoEngine method*), 22
detect_db_version() (*pwman.data.convertdb.DBConverter static method*), 14
do_cls() (*pwman.ui.baseui.BaseCommands method*), 16
do_cls() (*pwman.ui.win.PwmanCliWin method*), 19
do_copy() (*pwman.ui.baseui.BaseCommands method*), 16
do_copy() (*pwman.ui.mac.PwmanCliMac method*), 18
do_copy() (*pwman.ui.win.PwmanCliWin method*), 19
do_cp() (*pwman.ui.baseui.AliasesMixin method*), 16
do_delete() (*pwman.ui.baseui.BaseCommands method*), 17

do_e() (*pwman.ui.baseui.AliasesMixin method*), 16
do_edit() (*pwman.ui.baseui.BaseCommands method*), 17
do_EOF() (*pwman.ui.baseui.AliasesMixin method*), 16
do_EOF() (*pwman.ui.baseui.BaseCommands method*), 16
do_exit() (*pwman.ui.baseui.BaseCommands method*), 17
do_export() (*pwman.ui.baseui.BaseCommands method*), 17
do_forget() (*pwman.ui.baseui.BaseCommands method*), 17
do_h() (*pwman.ui.baseui.AliasesMixin method*), 16
do_info() (*pwman.ui.baseui.BaseCommands method*), 17
do_l() (*pwman.ui.baseui.AliasesMixin method*), 16
do_list() (*pwman.ui.baseui.BaseCommands method*), 17
do_ls() (*pwman.ui.baseui.AliasesMixin method*), 16
do_n() (*pwman.ui.baseui.AliasesMixin method*), 16
do_new() (*pwman.ui.baseui.BaseCommands method*), 17
do_o() (*pwman.ui.baseui.AliasesMixin method*), 16
do_open() (*pwman.ui.baseui.BaseCommands method*), 17
do_p() (*pwman.ui.baseui.AliasesMixin method*), 16
do_passwd() (*pwman.ui.baseui.BaseCommands method*), 17
do_pp() (*pwman.ui.baseui.BaseCommands method*), 17
do_print() (*pwman.ui.baseui.BaseCommands method*), 17
do_print() (*pwman.ui.win.PwmanCliWin method*), 19
do_rm() (*pwman.ui.baseui.AliasesMixin method*), 16
do_tags() (*pwman.ui.baseui.BaseCommands method*), 17

E

editnode() (*pwman.data.database.Database method*), 14
encode_AES() (*in module pwman.util.crypto_engine*), 22
encrypt() (*pwman.util.crypto_engine.CryptoEngine method*), 22
error() (*pwman.ui.baseui.BaseCommands method*), 17
error() (*pwman.util.callback.Callback method*), 20

F

fetch_crypto_info() (*pwman.data.database.Database method*), 14
find_config_dir (*in module pwman.util.config*), 21
forget() (*pwman.util.crypto_engine.CryptoEngine method*), 22
from_encrypted_entries() (*pwman.data.nodes.Node class method*), 15

G

generate_password() (*in module pwman.util.crypto_engine*), 22
get() (*pwman.util.crypto_engine.CryptoEngine class method*), 22
get_cipher() (*in module pwman.util.crypto_engine*), 22
get_conf() (*in module pwman.__init__*), 13
get_conf_options() (*in module pwman.__init__*), 13
get_db_version() (*in module pwman.__init__*), 13
get_digest() (*in module pwman.util.crypto_engine*), 22
get_node() (*pwman.data.database.Database method*), 14
get_or_create_pass() (*in module pwman.ui.tools*), 19
get_pass_conf() (*in module pwman.util.config*), 22
get_salt_digest() (*pwman.util.crypto_engine.CryptoEngine method*), 22
get_ui_platform() (*in module pwman.ui.cli*), 18
get_user_password() (*pwman.data.database.Database method*), 14
get_value() (*pwman.util.config.Config method*), 21
getinput() (*in module pwman.ui.tools*), 19
getinput() (*pwman.ui.tools.CLICallback method*), 18
getinput() (*pwman.util.callback.Callback method*), 20
getinput() (*pwman.util.callback.CLICallback method*), 19
getnewsecret() (*pwman.ui.tools.CLICallback method*), 18
getnewsecret() (*pwman.util.callback.CLICallback method*), 20
getnodes() (*pwman.data.database.Database method*), 14
getsecret() (*pwman.ui.tools.CLICallback method*), 18
getsecret() (*pwman.util.callback.Callback method*), 20
getsecret() (*pwman.util.callback.CLICallback method*), 20
Green (*pwman.ui.tools.ANSI attribute*), 18

H

heardEnterWin() (*in module pwman.ui.win*), 19
help_cls() (*pwman.ui.baseui.HelpUIMixin method*), 17

help_copy() (*pwman.ui.baseui.HelpUIMixin method*), 17
help_delete() (*pwman.ui.baseui.HelpUIMixin method*), 17
help_edit() (*pwman.ui.baseui.HelpUIMixin method*), 17
help_exit() (*pwman.ui.baseui.HelpUIMixin method*), 17
help_export() (*pwman.ui.baseui.HelpUIMixin method*), 17
help_forget() (*pwman.ui.baseui.HelpUIMixin method*), 17
help_help() (*pwman.ui.baseui.HelpUIMixin method*), 17
help_info() (*pwman.ui.baseui.HelpUIMixin method*), 17
help_list() (*pwman.ui.baseui.HelpUIMixin method*), 17
help_new() (*pwman.ui.baseui.HelpUIMixin method*), 17
help_open() (*pwman.ui.baseui.HelpUIMixin method*), 17
help_passwd() (*pwman.ui.baseui.HelpUIMixin method*), 17
help_print() (*pwman.ui.baseui.HelpUIMixin method*), 18
help_tags() (*pwman.ui.baseui.HelpUIMixin method*), 18
HelpUIMixin (*class in pwman.ui.baseui*), 17

I

Importer (*class in pwman.exchange.importer*), 16
invoke_converter() (*pwman.data.convertdb.DBConverter static method*), 14
is_latest_version() (*in module pwman.__init__*), 13

L

lazy_get_nodes() (*pwman.data.database.Database method*), 14
lazy_list_node_ids() (*pwman.data.database.Database method*), 14
listtags() (*pwman.data.database.Database method*), 14
loadkey() (*pwman.data.database.Database method*), 14

M

Magenta (*pwman.ui.tools.ANSI attribute*), 18
main() (*in module pwman.ui.cli*), 18

N

no_parse_uri() (*in module pwman.data.factory*), 15
Node (*class in pwman.data.nodes*), 15
notes (*pwman.data.nodes.Node attribute*), 15
notice() (*pwman.util.callback.Callback method*), 20

O

open() (*pwman.data.database.Database method*), 14
open_url() (*in module pwman.ui.tools*), 19

P

parse_postgres_uri() (*in module pwman.data.factory*), 15
parse_sqlite_uri() (*in module pwman.data.factory*), 15
parser_options() (*in module pwman.__init__*), 13
password (*pwman.data.nodes.Node attribute*), 15
PkgMetadata (*class in pwman.__init__*), 13
prepare_data() (*in module pwman.util.crypto_engine*), 22
print_success() (*pwman.data.convertdb.DBConverter method*), 14
pwman.__init__ (*module*), 13
pwman.data.convertdb (*module*), 13
pwman.data.database (*module*), 14
pwman.data.drivers.sqlite (*module*), 15
pwman.data.factory (*module*), 15
pwman.data.nodes (*module*), 15
pwman.exchange.importer (*module*), 16
pwman.ui (*module*), 16
pwman.ui.baseui (*module*), 16
pwman.ui.cli (*module*), 18
pwman.ui.mac (*module*), 18
pwman.ui.tools (*module*), 18
pwman.ui.win (*module*), 19
pwman.util.callback (*module*), 19
pwman.util.config (*module*), 20
pwman.util.crypto_engine (*module*), 22
PwmanCli (*class in pwman.ui.cli*), 18
PwmanCliMac (*class in pwman.ui.mac*), 18
PwmanCliWin (*class in pwman.ui.win*), 19
PwmanConvertKey (*class in pwman.data.convertdb*), 14

R

read_old_db() (*pwman.data.convertdb.DBConverter method*), 14
read_old_db() (*pwman.data.convertdb.PwmanConvertKey method*), 14
Red (*pwman.ui.tools.ANSI attribute*), 18
removenodes() (*pwman.data.database.Database method*), 14
Reset (*pwman.ui.tools.ANSI attribute*), 18

run() (*pwman.data.convertdb.DBConverter method*), 14
run() (*pwman.exchange.importer.CSVImporter method*), 16
run() (*pwman.exchange.importer.Importer method*), 16
run() (*pwman.ui.tools.CMDLoop method*), 19

S

save() (*pwman.util.config.Config method*), 21
save_crypto_info() (*pwman.data.database.Database method*), 14
save_new_nodes_to_db() (*pwman.data.convertdb.DBConverter method*), 14
save_old_key() (*pwman.data.convertdb.DBConverter method*), 14
save_old_key() (*pwman.data.convertdb.PwmanConvertKey method*), 14
savekey() (*pwman.data.database.Database method*), 15
set_db() (*in module pwman.__init__*), 13
set_salt_digest() (*pwman.util.crypto_engine.CryptoEngine method*), 22
set_selection() (*in module pwman.ui.tools*), 19
set_umask() (*in module pwman.__init__*), 13
set_value() (*pwman.util.config.Config method*), 21
set_xsel() (*in module pwman.__init__*), 13
SQLite (*class in pwman.data.drivers.sqlite*), 15

T

tags (*pwman.data.nodes.Node attribute*), 15
text_to_clipboards() (*in module pwman.ui.tools*), 19
text_to_mcclipboard() (*in module pwman.ui.tools*), 19
to_encdict() (*pwman.data.nodes.Node method*), 15
typeset() (*in module pwman.ui.tools*), 19

U

Underscore (*pwman.ui.tools.ANSI attribute*), 18
undoc_header (*pwman.ui.cli.PwmanCli attribute*), 18
url (*pwman.data.nodes.Node attribute*), 15
username (*pwman.data.nodes.Node attribute*), 15

W

warning() (*pwman.util.callback.Callback method*), 20
White (*pwman.ui.tools.ANSI attribute*), 18
winGetClipboard() (*in module pwman.ui.win*), 19
winSetClipboard() (*in module pwman.ui.win*), 19

Y

Yellow (*pwman.ui.tools.ANSI attribute*), 18

Z

zerome() (*in module pwman.util.crypto_engine*), 22