k2hash

Release 1.0.0

Hirotaka Wakabayashi, Takeshi Nakatani

Mar 03, 2022

CONTENTS:

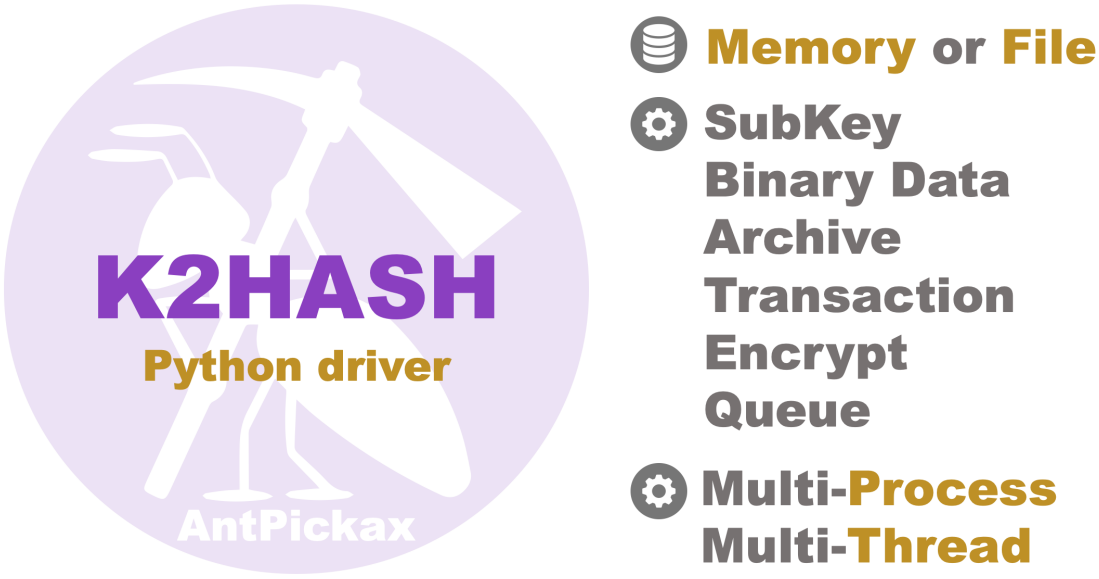
CHAPTER

ONE

K2HASH_PYTHON

1.1 Overview

k2hash_python is an official python driver for k2hash.



1.2 Install

Firstly you must install the k2hash shared library:

```
curl -o- https://raw.github.com/yahoojapan/k2hash_python/master/utils/libk2hash.sh | bash
```

Then, let's install k2hash using pip:

```
pip install k2hash
```

1.3 Usage

Try to set a key and get it:

```
import k2hash

k = k2hash.K2hash('test.k2h')
k.set('hello', 'world')
v = k.get('hello')
print(v)    // world
```

1.4 Development

Clone this repository and go into the directory, then run the following command:

```
$ python3 -m pip install --upgrade build
$ python3 -m build
```

1.5 Documents

Here are documents including other components.

Document top page

About K2HASH

About AntPickax

1.6 Packages

Here are packages including other components.

k2hash(python packages)

1.7 License

MIT License. See the LICENSE file.

1.8 AntPickax

k2hash_python is a project by AntPickax, which is an open source team in Yahoo Japan Corporation.

CHAPTER

TWO

K2HASH

2.1 k2hash package

2.1.1 Submodules

2.1.2 k2hash.k2hash module

K2hash Python Driver under MIT License

class k2hash.k2hash.**K2hash**(*k2hfile=''*, *flag=None*, *readonly=True*, *removefile=True*, *fullmap=True*, *maskbit=8*, *cmaskbit=4*, *maxelementcnt=1024*, *pagesize=512*, *waitms=0*, *logfile=''*)

Bases: `object`

K2hash class provides methods to handle key/value pairs in k2hash hash database.

K2H_INVALID_HANDLE = 0

add_attribute_plugin_lib(*path*)
Adds a shared library that handles an attribute

add_decryption_password(*password*)
Adds a passphrase to decrypt a value.

add_subkey(*key*, *subkey*, *subval*, *password=None*, *expire_duration=None*, *time_unit=TimeUnit.SECONDS*)
Adds subkeys to a key/value pair.

begin_tx(*txfile*, *prefix=None*, *param=None*, *expire_duration=None*)
Starts a transaction logging.

close()
Closes a k2h file.

static create(*pathname*, *maskbit=8*, *cmaskbit=4*, *maxelementcnt=1024*, *pagesize=512*)
Creates a k2hash file.

dump_to_file(*path*, *is_skip_error=True*)
Dumps data to a file.

enable_encryption(*enable=True*)
Enables a feature to encrypt a value.

enable_history(*enable=True*)
Enables a feature to record a key modification history.

enable_mtime(*enable=True*)
Enables a feature to record value modification time.

get(*key*, *password=None*)
Gets the value

get_attributes(*key*, *use_str=True*)
Gets attributes of a key.

get_iterator(*key=None*)
Returns the k2hash iterator

get_subkeys(*key*, *use_str=True*)
Gets keys of subkeys of a key.

get_tx_file_fd()
Gets a transaction log file descriptor.

static get_tx_pool_size()
Gets the number of transaction thread pool.

property handle
Returns a Queue handle.

property libc
returns libc handle

property libk2hash
returns libk2hash handle

load_from_file(*path*, *is_skip_error=True*)
Loads data from a file.

print_attribute_plugins()
Prints attribute plugins to stderr.

print_attributes()
Prints attributes to stderr.

print_data_stats()
Prints data statistics.

print_table_stats(*level=DumpLevel.HEADER*)
Prints k2hash key table information.

remove(*key*, *remove_all_subkeys=False*)
Removes a key.

remove_subkeys(*key*, *subkeys*)
Removes subkeys from the key.

rename(*key*, *newkey*)
Renames a key with a new key.

set(*key*, *val*, *password=None*, *expire_duration=None*, *time_unit=TimeUnit.SECONDS*)
Sets a key/value pair

set_attribute(*key*, *attr_name*, *attr_val*)
Sets an attribute of a key.

set_default_encryption_password(*password*)
Sets the default encryption passphrase.

set_encryption_password_file(*path*)
Sets the data encryption password file.

set_expiration_duration(*expire_duration*, *time_unit=TimeUnit.SECONDS*)
Sets the duration to expire a value.

set_log_level(*level=LogLevel.INFO*)
Creates a k2hash file.

set_subkeys(*key*, *subkeys*, *password=None*, *expire_duration=None*, *time_unit=TimeUnit.SECONDS*)
Sets subkeys.

static set_tx_pool_size(*size*)
Sets the number of transaction thread pool.

stop_tx()
Stops a transaction logging.

static version()
Prints version information.

class k2hash.k2hash.**K2hashIterator**(*k2h*, *key=None*)
Bases: object

implements iterator of k2hash

2.1.3 k2hash.keyqueue module

K2hash Python Driver under MIT License

class k2hash.keyqueue.**KeyQueue**(*k2h*, *fifo=True*, *prefix=None*, *password=None*, *expire_duration=None*)
Bases: object

KeyQueue class provides methods to handle key/value pairs in k2hash hash database.

clear()
Removes all of the elements from this collection (optional operation).

close()
Free QueueHandle

element(*position=0*)
Finds and gets a object from the head of this queue.

empty()
Returns true if, and only if, queue size is 0.

get()
Finds and gets a object from the head of this queue.

property handle
Returns a Queue handle.

print()
Print the objects in this queue.

put(*obj*)
Inserts an element into the tail of this queue.

qsize()
Returns the number of queue.

remove(*count=1*)
Removes objects from this queue.

2.1.4 k2hash.queue module

K2hash Python Driver under MIT License

class k2hash.queue.**Queue**(*k2h*, *fifo=True*, *prefix=None*, *password=None*, *expire_duration=None*)

Bases: object

Queue class provides methods to handle key/value pairs in k2hash hash database.

clear()
Removes all of the elements from this collection (optional operation).

close()
Free QueueHandle

element(*position=0*)
Finds and gets a object from the head of this queue.

empty()
Returns true if, and only if, queue size is 0.

get()
Finds and gets a object from the head of this queue.

property handle
Returns a Queue handle.

print()
Print the objects in this queue.

put(*obj*, *attrs=None*)
Inserts an element into the tail of this queue.

qsize()
Returns the number of queue.

remove(*count=1*)
Removes objects from this queue.

2.1.5 Module contents

k2hash package

class k2hash.**AttrPack**

Bases: _ctypes.Structure

C Attr structure

keylength
Structure/Union member

pkey
Structure/Union member

pval
Structure/Union member

vallength
Structure/Union member

class k2hash.**DumpLevel**(*value*)

Bases: enum.Enum

k2hash file status information

ELEMENT = 4

HASH_TABLE = 2

HEADER = 1

PAGE = 5

SUB_HASH_TABLE = 3

class k2hash.**K2hash**(*k2hfile=''*, *flag=None*, *readonly=True*, *removefile=True*, *fullmap=True*, *maskbit=8*, *cmaskbit=4*, *maxelementcnt=1024*, *pagesize=512*, *waitms=0*, *logfile=''*)

Bases: `object`

K2hash class provides methods to handle key/value pairs in k2hash hash database.

K2H_INVALID_HANDLE = 0

add_attribute_plugin_lib(*path*)
Adds a shared library that handles an attribute

add_decryption_password(*password*)
Adds a passphrase to decrypt a value.

add_subkey(*key*, *subkey*, *subval*, *password=None*, *expire_duration=None*, *time_unit=TimeUnit.SECONDS*)
Adds subkeys to a key/value pair.

begin_tx(*txfile*, *prefix=None*, *param=None*, *expire_duration=None*)
Starts a transaction logging.

close()
Closes a k2h file.

static create(*pathname*, *maskbit=8*, *cmaskbit=4*, *maxelementcnt=1024*, *pagesize=512*)
Creates a k2hash file.

dump_to_file(*path*, *is_skip_error=True*)
Dumps data to a file.

enable_encryption(*enable=True*)
Enables a feature to encrypt a value.

enable_history(*enable=True*)
Enables a feature to record a key modification history.

enable_mtime(*enable=True*)
Enables a feature to record value modification time.

get(*key*, *password=None*)
Gets the value

get_attributes(*key*, *use_str=True*)
Gets attributes of a key.

get_iterator(*key=None*)
Returns the k2hash iterator

get_subkeys(*key*, *use_str=True*)
Gets keys of subkeys of a key.

get_tx_file_fd()
Gets a transaction log file descriptor.

static get_tx_pool_size()
Gets the number of transaction thread pool.

property handle
Returns a Queue handle.

property libc
returns libc handle

property libk2hash
returns libk2hash handle

load_from_file(*path*, *is_skip_error=True*)
Loads data from a file.

print_attribute_plugins()
Prints attribute plugins to stderr.

print_attributes()
Prints attributes to stderr.

print_data_stats()
Prints data statistics.

print_table_stats(*level=DumpLevel.HEADER*)
Prints k2hash key table information.

remove(*key*, *remove_all_subkeys=False*)
Removes a key.

remove_subkeys(*key*, *subkeys*)
Removes subkeys from the key.

rename(*key*, *newkey*)
Renames a key with a new key.

set(*key*, *val*, *password=None*, *expire_duration=None*, *time_unit=TimeUnit.SECONDS*)
Sets a key/value pair

set_attribute(*key*, *attr_name*, *attr_val*)
Sets an attribute of a key.

set_default_encryption_password(*password*)
Sets the default encryption passphrase.

set_encryption_password_file(*path*)
Sets the data encryption password file.

set_expiration_duration(*expire_duration*, *time_unit=TimeUnit.SECONDS*)
Sets the duration to expire a value.

set_log_level(*level=LogLevel.INFO*)
Creates a k2hash file.

set_subkeys(*key*, *subkeys*, *password=None*, *expire_duration=None*, *time_unit=TimeUnit.SECONDS*)
Sets subkeys.

static set_tx_pool_size(*size*)
Sets the number of transaction thread pool.

stop_tx()
Stops a transaction logging.

static version()
Prints version information.

class k2hash.**K2hashIterator**(*k2h*, *key=None*)
Bases: `object`

implements iterator of k2hash

class k2hash.**KeyPack**
Bases: `_ctypes.Structure`

C KeyPack structure

length
Structure/Union member

pkey
Structure/Union member

class k2hash.**KeyQueue**(*k2h*, *fifo=True*, *prefix=None*, *password=None*, *expire_duration=None*)
Bases: `object`

KeyQueue class provides methods to handle key/value pairs in k2hash hash database.

clear()
Removes all of the elements from this collection (optional operation).

close()
Free QueueHandle

element(*position=0*)
Finds and gets a object from the head of this queue.

empty()
Returns true if, and only if, queue size is 0.

get()
Finds and gets a object from the head of this queue.

property handle
Returns a Queue handle.

print()
Print the objects in this queue.

put(*obj*)
Inserts an element into the tail of this queue.

qsize()
Returns the number of queue.

remove(*count=1*)
Removes objects from this queue.

class k2hash.**LogLevel**(*value*)
Bases: `enum.Enum`

k2hash log level

DEBUG = 5

ERROR = 2

INFO = 4

SILENT = 1

WARNING = 3

class k2hash.**OpenFlag**(*value*)
Bases: enum.Enum

k2hash file open flags

EDIT = 2

MEMORY = 4

READ = 1

TEMPFILE = 3

class k2hash.**Queue**(*k2h*, *fifo=True*, *prefix=None*, *password=None*, *expire_duration=None*)
Bases: object

Queue class provides methods to handle key/value pairs in k2hash hash database.

clear()
Removes all of the elements from this collection (optional operation).

close()
Free QueueHandle

element(*position=0*)
Finds and gets a object from the head of this queue.

empty()
Returns true if, and only if, queue size is 0.

get()
Finds and gets a object from the head of this queue.

property handle
Returns a Queue handle.

print()
Print the objects in this queue.

put(*obj*, *attrs=None*)
Inserts an element into the tail of this queue.

qsize()
Returns the number of queue.

remove(*count=1*)
Removes objects from this queue.

class k2hash.**TimeUnit**(*value*)
Bases: enum.Enum

k2hash time units

DAYS = 1

HOURS = 2

MILLISECONDS = 3

MINUTES = 4

SECONDS = 5

CHAPTER

THREE

CREDITS

3.1 Development Lead

- Hirotaka Wakabayashi <hiwakaba@yahoo-corp.jp>

3.2 Contributors

- Takeshi Nakatani <ggtakec@gmail.com>

CHAPTER

FOUR

HISTORY

4.1 1.0.0 (2022-02-07)

- First release on PyPI.

CHAPTER

FIVE

INDICES AND TABLES

- genindex
- modindex
- search

PYTHON MODULE INDEX

k

INDEX

I

K

L

M

O

P

Q

R

S

T

V

W