k2hdkc

Release 1.0.0

Hirotaka Wakabayashi, Takeshi Nakatani

Mar 04, 2022

CONTENTS:

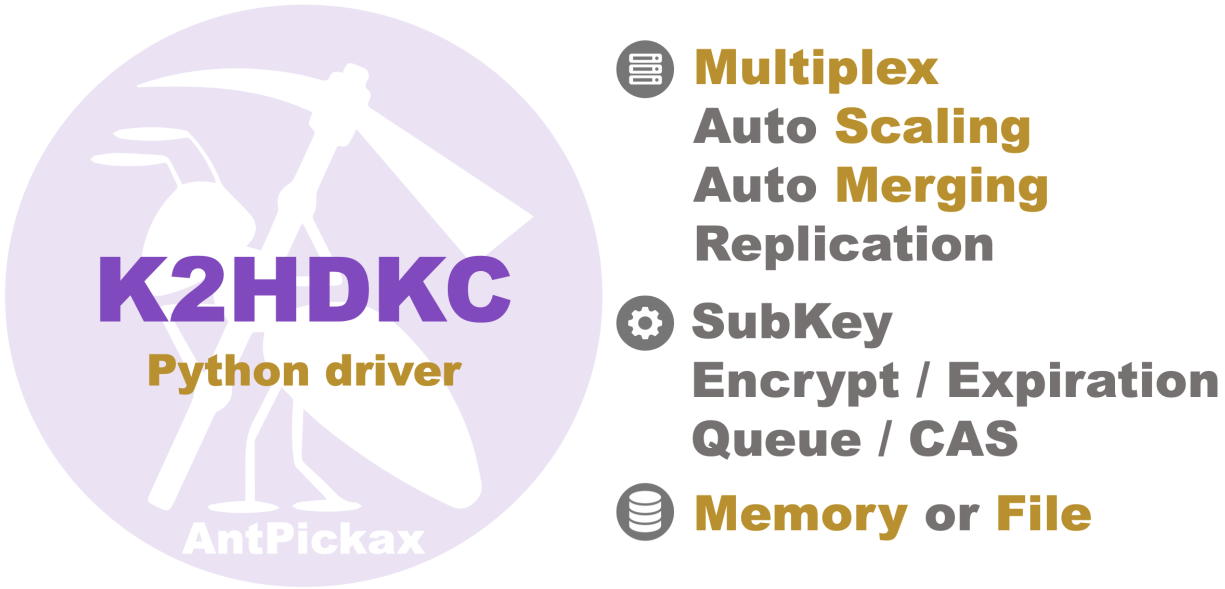
CHAPTER

ONE

K2HDKC_PYTHON

1.1 Overview

k2hdkc_python is an official python driver for k2hdkc.



1.2 Install

Let's install k2hdkc using pip:

```
pip install k2hdkc
```

1.3 Usage

Firstly you must install the k2hdkc shared library:

```
$ curl -o- https://raw.github.com/yahoojapan/k2hdkc_python/master/cluster/start_server.
↪sh | bash
```

Then, Let's try to set a key and get it:

```
import k2hdkc

k = k2hdkc.K2hdkc('slave.yaml')
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 K2HDKC

About AntPickax

1.6 Packages

Here are packages including other components.

k2hdkc(python packages)

1.7 License

MIT License. See the LICENSE file.

1.8 AntPickax

k2hdkc_python is a project by AntPickax, which is an open source team in Yahoo Japan Corporation.

CHAPTER

TWO

K2HDKC

2.1 k2hdkc package

2.1.1 Submodules

2.1.2 k2hdkc.k2hdkc module

K2hdkc Python Driver

class k2hdkc.k2hdkc.**K2hdkc**(*conf_file*, *port=8031*, *cuk=None*, *rejoin=True*, *rejoin_forever=True*, *clear_backup=True*)

Bases: `object`

K2hdkc class provides methods to handle key/value pairs in k2hdkc hash database.

K2H_INVALID_HANDLE = 0

add_subkey(*key*, *subkey*, *subval*, *check_attr=True*, *password=None*, *expire_duration=None*, *time_unit=TimeUnit.SECONDS*)

Adds a new subkey to a current subkey.

cas_decrement(*key*, *password=None*, *expire_duration=None*)

Decrements a variable in a cluster by using a CAS operation.

cas_get(*key*, *data_type*, *password=None*, *expire_duration=None*)

Gets a variable from a cluster using a CAS operation.

cas_increment(*key*, *password=None*, *expire_duration=None*)

Increments a variable in a cluster by using a CAS operation.

cas_init(*key*, *val=None*, *password=None*, *expire_duration=None*)

Initializes a variable in a cluster by using a CAS operation.

cas_set(*key*, *old_val*, *new_val*, *password=None*, *expire_duration=None*)

Sets a value in a cluster by using a CAS operation.

clear_subkeys(*key*)

Clears subkeys of a key. Another subkeys that a subkey has will be removed recursively.

close()

Closes the handle

get(*key*, *password=None*)

Gets the value

get_attributes(*key*, *use_str=True*)

Retrievs attributes of a key.

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

keyqueue_get(*prefix*, *is_fifo=True*, *password=None*, *expire_duration=None*)
Gets a new key/value element from queue.

keyqueue_put(*prefix*, *key*, *val*, *is_fifo=True*, *is_check_attr=True*, *password=None*, *expire_duration=None*)
Adds a new key/value pair element to a queue.

property libc
returns libc handle

property libk2hdkc
returns libk2hkc handle

queue_get(*prefix*, *is_fifo=True*, *password=None*, *expire_duration=None*)
Gets a new element to a queue.

queue_put(*prefix*, *val*, *is_fifo=True*, *is_check_attr=True*, *password=None*, *expire_duration=None*)
Adds a new element to a queue.

remove(*key*)
Removes a key from a cluster.

remove_subkeys(*key*, *subkeys*, *nested=False*)
Removes a subkey from the current subkeys.

rename(*key*, *newkey*, *parent_key=None*, *is_check_attr=True*, *password=None*, *expire_duration=None*)
Renames a key in a cluster.

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

set_subkeys(*key*, *subkeys*)
Replaces current subkeys with new one.

2.1.3 Module contents

k2hdkc package

class k2hdkc.**K2hdkc**(*conf_file*, *port=8031*, *cuk=None*, *rejoin=True*, *rejoin_forever=True*, *clear_backup=True*)
Bases: `object`

K2hdkc class provides methods to handle key/value pairs in k2hdkc hash database.

K2H_INVALID_HANDLE = 0

add_subkey(*key*, *subkey*, *subval*, *check_attr=True*, *password=None*, *expire_duration=None*, *time_unit=TimeUnit.SECONDS*)
Adds a new subkey to a current subkey.

cas_decrement(*key*, *password=None*, *expire_duration=None*)
Decrements a variable in a cluster by using a CAS operation.

cas_get(*key*, *data_type*, *password=None*, *expire_duration=None*)
Gets a variable from a cluster using a CAS operation.

cas_increment(*key*, *password=None*, *expire_duration=None*)
Increments a variable in a cluster by using a CAS operation.

cas_init(*key*, *val=None*, *password=None*, *expire_duration=None*)
Initializes a variable in a cluster by using a CAS operation.

cas_set(*key*, *old_val*, *new_val*, *password=None*, *expire_duration=None*)
Sets a value in a cluster by using a CAS operation.

clear_subkeys(*key*)
Clears subkeys of a key. Another subkeys that a subkey has will be removed recursively.

close()
Closes the handle

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

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

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

keyqueue_get(*prefix*, *is_fifo=True*, *password=None*, *expire_duration=None*)
Gets a new key/value element from queue.

keyqueue_put(*prefix*, *key*, *val*, *is_fifo=True*, *is_check_attr=True*, *password=None*, *expire_duration=None*)
Adds a new key/value pair element to a queue.

property libc
returns libc handle

property libk2hdkc
returns libk2hkc handle

queue_get(*prefix*, *is_fifo=True*, *password=None*, *expire_duration=None*)
Gets a new element to a queue.

queue_put(*prefix*, *val*, *is_fifo=True*, *is_check_attr=True*, *password=None*, *expire_duration=None*)
Adds a new element to a queue.

remove(*key*)
Removes a key from a cluster.

remove_subkeys(*key*, *subkeys*, *nested=False*)
Removes a subkey from the current subkeys.

rename(*key*, *newkey*, *parent_key=None*, *is_check_attr=True*, *password=None*, *expire_duration=None*)
Renames a key in a cluster.

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

set_subkeys(*key*, *subkeys*)
Replaces current subkeys with new one.

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