moecache Documentation

Release 1.0

Rackspace

April 08, 2016

Contents:

CHAPTER 1

moecache Module

A memcached client with a different shading strategy.

Usage example:

```python
import moecache

with moecache.Client([("127.0.0.1", 11211), ("127.0.0.1", 11213)],
                     timeout=1, connect_timeout=5) as mc:
    mc.set("some_key", "Some value")
    value = mc.get("some_key")
    mc.delete("another_key")
```

Note: If the value to store is of type `str`, the entry is binary compatible with EnyimMemcached; otherwise, the value will be "pickled" with protocol version 2 and can only be read by moecache.

class `moecache.`**`Client`**(*servers*[, *timeout*[, *connect_timeout*]])

Creates an object to hold a moecache session. The object is also a context manager, which automatically closes the sockets when the control leaves the `with` statement.

`servers` can be a single server, or a list of servers, where each server is a `(host, port)` tuple, same as a `socket` AF_INET address.

If `timeout` is not specified, socket operations may block forever. If `connect_timeout` is not specified, the `timeout` setting will also be applied to the socket `connect()` operations.

`set`(*key*, *val*, *exptime=0*)

Sets a key to a value with an optional expire time in seconds (0 means don't auto-expire).

A valid `key` is a string with a minimal length of 1 and a maximal length of 250, and each character is an ASCII graph character (printable except spaces).

A valid `exptime` is a non-negative integer.

If any of these arguments is invalid, `ValidationException` will be raise. May also raise `ClientException` and socket errors.

`get`(*key*)

Gets a single value. Returns `None` if the key does not exist.

Raises `ValidationException` if `key` is invalid. May also raise `ClientException` and socket errors.

`delete`(*key*)

Deletes a key/value pair.

Raises `ValidationException` if `key` is invalid. May also raise `ClientException` and socket errors.

Note: The postcondition of this operation is that the entry no longer exists, so if the key does not exists at the first place, nothing happens and the function returns without error.

`stats`([*additional_args*])

Aggregates the stats from all the servers.

`additional_args` is a byte string being passed verbatim to the servers. See the memcached wiki for details or the spec for even more details.

Raises `ClientException` and socket errors.

`close`()

Closes any opened socket.

Sockets are automatically closed when the `Client` object gets out of the context.

Raises socket errors.

exception `moecache.`**`ClientException`**(*msg*[, *item*])

Raised when memcached does something we don't expect, or the memcached deployment is not compatible with moecache.

Note: This does not include socket errors.

exception `moecache.`**`ValidationException`**(*msg*, *item*)

Bases: *`moecache.ClientException`*

Raised when the user input is invalid to this library.

CHAPTER 2

Indices and tables

- genindex
- modindex
- search

Python Module Index

Index