etc Documentation

Release 0.0.0-dev

Heungsub Lee

March 01, 2016

Contents

`etc` is an etcd Python client library. It provides all etcd options as *snake_case*. So there's no *camelCase* confusion. It also provides several useful sugar functions such as `etc.keep_node()`:

```
import etc

etcd = etc.etcd('http://127.0.0.1:4001')

try:
    etcd.set('/etc', u'what!', prev_value=u'what?', ttl=42)
except etc.TestFailed:
    pass
if isinstance(etcd.wait('/etc'), etc.Expired):
    print 'Expired'
```

CHAPTER 1

Installation

`etc` is not available in PyPI yet. You should install via GitHub:

```
$ pip install https://github.com/sublee/etc/archive/master.zip
```

CHAPTER 2

Usage

First of all, create a client object with your etcd URL:

```
import etc
etcd = etc.etcd('http://127.0.0.1:4001')
```

All etcd methods are in the client. *etc.Client.get()*, *etc.Client.set()*, and *etc.Client.delete()* are basic methods for most cases:

```
>>> etcd.set('/hello', u'Hello, world')
<etc.Set <etc.Value /hello='Hello, world' ...> ...>
>>> etcd.get('/hello')
<etc.Got <etc.Value /hello='Hello, world' ...> ...>
>>> etcd.delete('/hello')
<etc.Deleted ... prev_node=<etc.Value /hello='Hello, world' ...> ...>
>>> etcd.get('/hello')
Traceback (most recent call last):
  ...
etc.errors.KeyNotFound: [100] Key not found (/hello)
```

All etcd result types are mapped with subclasses of *etc.EtcdResult*; `etc.Got`, `etc.Set`, `etc.Deleted`, `etc.Created`, `etc.Updated`, `etc.Expired`, `etc.ComparedThenSwapped`, `etc.ComparedThenDeleted`. A result contains a node which is an instance of *etc.Node*. There're 2 subclasses; *etc.Value* and *etc.Directory*. You will check whether a node is a directory or not by `isinstance()`:

```
isinstance(etcd.get('/etc').node, etc.Value)
```

A directory node can be defined by *dir* parameter:

```
>>> etcd.set('/container', dir=True)
<etc.Set <etc.Directory /container[0] ...> ...>
```

CHAPTER 3

API

etc.**etcd**(*url='http://127.0.0.1:4001'*, *mock=False*, ***kwargs*)
Creates an etcd client.

class etc.**Client**(*adapter*)
An etcd client. It wraps an etc.adapter.Adapter and exposes humane public methods.

get(*key*, *recursive=False*, *sorted=False*, *quorum=False*, *timeout=None*)
Gets a value of a node.

```
>>> etcd.get('/hello')
<etc.Got <etc.Value /hello='Hello, world' ...> ...>
>>> etcd.get('/container', recursive=True)
<etc.Got <etc.Directory /container[2] ...> ...>
```

Parameters

- **key** – the key of the node to get.
- **recursive** – include sub nodes recursively.
- **sorted** – sort sub nodes by their keys.
- **quorum** – ensure all quorums are ready to make result.
- **timeout** – timeout in seconds.

wait(*key*, *index=0*, *recursive=False*, *sorted=False*, *quorum=False*, *timeout=None*)
Waits until a node changes.

Parameters

- **key** – the key of the node where it waits changes from.
- **index** – wait a modification after this index.
- **recursive** – wait until sub nodes change also.
- **sorted** – sort sub nodes by their keys.
- **quorum** – ensure all quorums are ready to make result.
- **timeout** – timeout in seconds.

set(*key*, *value=None*, *dir=False*, *ttl=None*, *prev_value=None*, *prev_index=None*, *timeout=None*)
Sets a value to a node.

Parameters

- **key** – the key of the node to be created or updated.
- **value** (*unicode*) – the node value. This parameter and *dir* parameter are exclusive of each other.
- **dir** – make the node to be a directory. This parameter and *value* parameter are exclusive of each other.
- **ttl** – tile to the node lives in seconds.
- **prev_value** – check the current node value is equivalent with this value.
- **prev_index** – check the current node's modified index is equivalent with this index.
- **timeout** – timeout in seconds.

create(*key*, *value=None*, *dir=False*, *ttl=None*, *timeout=None*)
Creates a new key.

Parameters

- **key** – the key of the node to be created.
- **value** (*unicode*) – the node value. This parameter and *dir* parameter are exclusive of each other.
- **dir** – make the node to be a directory. This parameter and *value* parameter are exclusive of each other.
- **ttl** – tile to the node lives in seconds.
- **timeout** – timeout in seconds.

update(*key*, *value=None*, *dir=False*, *ttl=None*, *prev_value=None*, *prev_index=None*, *timeout=None*)
Updates an existing key.

Parameters

- **key** – the key of the node to be updated.
- **value** (*unicode*) – the node value. This parameter and *dir* parameter are exclusive of each other.
- **dir** – make the node to be a directory. This parameter and *value* parameter are exclusive of each other.
- **ttl** – tile to the node lives in seconds.
- **prev_value** – check the current node value is equivalent with this value.
- **prev_index** – check the current node's modified index is equivalent with this index.
- **timeout** – timeout in seconds.

append(*key*, *value=None*, *dir=False*, *ttl=None*, *timeout=None*)
Creates a new automatically increasing key in the given directory key.

Parameters

- **key** – the directory node key which will contain the new node.
- **value** (*unicode*) – the node value. This parameter and *dir* parameter are exclusive of each other.
- **dir** – make the node to be a directory. This parameter and *value* parameter are exclusive of each other.
- **ttl** – tile to the node lives in seconds.

- **timeout** – timeout in seconds.

delete(*key*, *dir=False*, *recursive=False*, *prev_value=None*, *prev_index=None*, *timeout=None*)
Deletes a node.

Parameters

- **key** – the node key to be deleted.
- **dir** – whether the node is directory or not.
- **recursive** – delete sub nodes recursively.
- **prev_value** – check the current node value is equivalent with this value.
- **prev_index** – check the current node's modified index is equivalent with this index.
- **timeout** – timeout in seconds.

3.1 etcd Results

class etc.**Node**(*key*, *modified_index=None*, *created_index=None*, *ttl=None*, *expiration=None*)

class etc.**Value**(*key*, *value*, **args*, ***kwargs*)
An etcd value Node.

class etc.**Directory**(*key*, *nodes=()*, **args*, ***kwargs*)
An etcd directory Node.

class etc.**EtcdResult**(*node*, *prev_node=None*, *etcd_index=None*, *raft_index=None*, *raft_term=None*)
A successful etcd result.

Don't use this class directly. There're specific subclasses to be used instead.

Subclasses Got, Set, Deleted, Created, Updated, Expired, ComparedThenSwapped, ComparedThenDeleted.

class etc.**EtcdError**(*message=None*, *cause=None*, *index=None*)
A failed etcd result.

Subclasses KeyNotFound, TestFailed, NotFile, NoMorePeer, NotDir, NodeExist, KeyIsPreserved, RootROnly, DirNotEmpty, ExistingPeerAddr, Unauthorized, ValueRequired, PrevValueRequired, TTLNaN, IndexNaN, ValueOrTTLRequired, TimeoutNaN, NameRequired, IndexOrValueRequired, IndexValueMutex, InvalidField, InvalidForm, RaftInternal, LeaderElect, WatcherCleared, EventIndexCleared, StandbyInternal, InvalidActiveSize, InvalidRemoveDelay.

CHAPTER 4

Licensing and Author

This project is licensed under the BSD license. See LICENSE for the details.

I'm Heungsub Lee, a game server architect. Any regarding questions or patches are welcomed.

A

C

D

E

G

N

S

U

V

W