
pyetcd Documentation

Release 1.11.1

TwinDB Development Team

Jun 25, 2018

Contents

1	pyetcd	3
1.1	Features	3
1.2	Credits	3
2	Installation	5
2.1	Stable release	5
2.2	From sources	5
3	Usage	7
4	pyetcd	9
4.1	pyetcd package	9
5	Contributing	17
5.1	Types of Contributions	17
5.2	Get Started!	18
5.3	Pull Request Guidelines	19
5.4	Tips	19
6	Credits	21
6.1	Development Lead	21
6.2	Contributors	21
7	History	23
7.1	0.1.0 (2016-09-17)	23
8	Indices and tables	25
	Python Module Index	27

Contents:

Python library to work with Etcd.

- Free software: Apache Software License 2.0
- Documentation: <https://pyetcd.readthedocs.io>.

1.1 Features

- Basic key operations

1.2 Credits

This package was created with [Cookiecutter](#) and the [audreyr/cookiecutter-pypackage](#) project template.

2.1 Stable release

To install pyetcd, run this command in your terminal:

```
$ pip install pyetcd
```

This is the preferred method to install pyetcd, as it will always install the most recent stable release.

If you don't have [pip](#) installed, this [Python installation guide](#) can guide you through the process.

2.2 From sources

The sources for pyetcd can be downloaded from the [Github repo](#).

You can either clone the public repository:

```
$ git clone git://github.com/twindb/pyetcd
```

Or download the [tarball](#):

```
$ curl -OL https://github.com/twindb/pyetcd/tarball/master
```

Once you have a copy of the source, you can install it with:

```
$ python setup.py install
```


CHAPTER 3

Usage

Connect to local etcd instance. Write a key and read it after:

```
from pyetcd.client import Client

client = Client()
client.write('/message', 'Hello world')
response = client.read('/message')

print(response.node['value'])
```

Write and read from a remote etcd cluster:

```
from pyetcd.client import Client

client = Client([
    '10.0.1.10',
    '10.0.1.11',
    '10.0.1.12'
])
client.write('/message', 'Hello world')
response = client.read('/message')

print(response.node['value'])
```

Watch a key:

```
from pyetcd.client import Client

client = Client([
    '10.0.1.10',
    '10.0.1.11',
    '10.0.1.12'
])
client.write('/message', 'Hello world')
```

(continues on next page)

(continued from previous page)

```
response = client.read('/message', wait=True)
print(response.node['value'])
```

4.1 pyetcd package

4.1.1 Submodules

4.1.2 pyetcd.client module

module to connect to an etcd node and perform low rest API requests.

class `pyetcd.client.Client` (***kwargs*)

Bases: `object`

Etcd Client class.

Parameters *kwargs* – Keyword arguments:

- **host** (*str*, *list(str)*, *list(tuple)*) - **etcd node hostname** or list of hostnames or list of tuples (hostname, port). Default is '127.0.0.1'.
- **port** (*int*) - TCP port to connect to. Default is 2379.
- **srv_domain** (*str*) - Domain name if DNS discovery is used
- **version_prefix** (*str*) - API version prefix. Default is 'v2'.
- **allow_reconnect** (*bool*) - **If client fails to connect to** a cluster node connect to the next node in the cluster. Default is True.
- **protocol** (*str*) - **Protocol to connect to the cluster.** Default is 'http'.

Raises

- *ClientException* – if any errors
- *NotImplementedError* – if there is an attempt to use unsupported DNS discovery.

add_member (*peer_urls*)

Add a node to the cluster.

Parameters `peer_urls` – List of URL for inter-peer communication. For example:

```
["http://10.0.0.10:2380"]
```

Returns Information about the newly added node.

Return type *EtcResult*

compare_and_delete (*key*, *prev_value=None*, *prev_index=None*)

This command will delete a key only if the client-provided conditions are equal to the current conditions.

Parameters

- **key** – the key
- **prev_value** – checks the previous value of the key.
- **prev_index** – checks the previous modifiedIndex of the key.

Returns Result of operation.

Return type *EtcResult*

Raises

- *EtcException* – if etcd responds with error or HTTP error.
- *EtcNodeExist* – if any condition fails.

compare_and_swap (*key*, *value*, *prev_value=None*, *prev_index=None*, *prev_exist=None*, *ttl=None*)

This command will set the value of a key only if the client-provided conditions are equal to the current conditions.

Parameters

- **key** – key string
- **value** – key value
- **prev_value** – checks the previous value of the key.
- **prev_index** – checks the previous modifiedIndex of the key.
- **prev_exist** – checks existence of the key: if prevExist is True, it is an update request; if prevExist is False, it is a create request.
- **ttl** – set ttl on the key in seconds

Returns Result of operation.

Return type *EtcResult*

Raises

- *EtcException* – if etcd responds with error or HTTP error.
- *EtcNodeExist* – if condition `prev_exist=False` fails.
- *EtcTestFailed* – if condition `prev_value='bar'` fails.

delete (*key*)

Delete a key

Parameters **key** – Key

Returns Result of operation.

Return type *EtcResult*

Raises *EtcException* – if etcd responds with error or HTTP error

health

Returns True if the node is healthy

Return type bool

mkdir (*directory*)

Create directory

Parameters **directory** – string with directory name

Returns Result of operation.

Return type *EtcResult*

Raises *EtcException* – if etcd responds with error or HTTP error

read (*key*, ***kwargs*)

Read key value

Parameters **key** – Key

Returns Result of operation.

Return type *EtcResult*

Raises *EtcException* – if etcd responds with error or HTTP error

remove_member (*member_id*)

Remove a node from the cluster.

Parameters **member_id** – etcd identifier of the node. For example, 272e204152.

rmdir (*directory*, *recursive=False*)

Delete directory

Parameters

- **directory** – string with directory name
- **recursive** – recursively delete directory if not empty

Returns Result of operation.

Return type *EtcResult*

Raises *EtcException* – if etcd responds with error or HTTP error

update_ttl (*key*, *ttl*)

Update key's ttl

Parameters

- **key** – the key
- **ttl** – new ttl

Returns Result of operation.

Return type *EtcResult*

Raises

- *EtcException* – if etcd responds with error or HTTP error.
- *EtcKeyNotFound* – if the key doesn't exist

version()

Return Etcd server version

Returns string with Etcd server version. E.g. '2.3.7'

Return type str

version_cluster()

Return Etcd cluster version

Returns string with Etcd cluster version. E.g. '2.3.0'

Return type str

version_server()

Same as .version()

Returns string with Etcd server version. E.g. '2.3.7'

Return type str

write(key, value, ttl=None)

Write value to a key

Parameters

- **key** – Key
- **value** – Value
- **ttl** – Keys in etcd can be set to expire after a specified number of seconds. You can do this by setting a TTL (time to live) on the key.

Returns Result of operation.

Return type *EtcdResult*

Raises *EtcdException* – if etcd responds with error or HTTP error

exception `pyetcd.client.ClientException`

Bases: `exceptions.Exception`

Exception for errors in Client class

4.1.3 Module contents

pyetcd is a module to work with etcd cluster

exception `pyetcd.EtcdClientInternal`

Bases: *pyetcd.EtcdException*

Error ecodeClientInternal (http code 500)

exception `pyetcd.EtcdDirNotEmpty`

Bases: *pyetcd.EtcdException*

Error EcodeDirNotEmpty (http code 108)

exception `pyetcd.EtcdEmptyResponse`

Bases: *pyetcd.EtcdInvalidResponse*

Error that raises if response from etcd is empty

exception `pyetcd.EtcdEventIndexCleared`
Bases: `pyetcd.EtcdException`
Error EcodeEventIndexCleared (http code 401)

exception `pyetcd.EtcdException`
Bases: `exceptions.Exception`
Generic Etcd error.

exception `pyetcd.EtcdExistingPeerAddr`
Bases: `pyetcd.EtcdException`
Error ecodeExistingPeerAddr (http code 109)

exception `pyetcd.EtcdIndexNaN`
Bases: `pyetcd.EtcdException`
Error EcodeIndexNaN (http code 203)

exception `pyetcd.EtcdIndexOrValueRequired`
Bases: `pyetcd.EtcdException`
Error ecodeIndexOrValueRequired (http code 207)

exception `pyetcd.EtcdIndexValueMutex`
Bases: `pyetcd.EtcdException`
Error ecodeIndexValueMutex (http code 208)

exception `pyetcd.EtcdInvalidActiveSize`
Bases: `pyetcd.EtcdException`
Error ecodeInvalidActiveSize (http code 403)

exception `pyetcd.EtcdInvalidField`
Bases: `pyetcd.EtcdException`
Error EcodeInvalidField (http code 209)

exception `pyetcd.EtcdInvalidForm`
Bases: `pyetcd.EtcdException`
Error EcodeInvalidForm (http code 210)

exception `pyetcd.EtcdInvalidRemoveDelay`
Bases: `pyetcd.EtcdException`
Error ecodeInvalidRemoveDelay (http code 404)

exception `pyetcd.EtcdInvalidResponse`
Bases: `pyetcd.EtcdException`
Error that raises if response from etcd is invalid

exception `pyetcd.EtcdKeyIsPreserved`
Bases: `pyetcd.EtcdException`
Error ecodeKeyIsPreserved (http code 106)

exception `pyetcd.EtcdKeyNotFound`
Bases: `pyetcd.EtcdException`
Error EcodeKeyNotFound (http code 100)

exception `pyetcd.EtcdLeaderElect`

Bases: `pyetcd.EtcdException`

Error EcodeLeaderElect (http code 301)

exception `pyetcd.EtcdNameRequired`

Bases: `pyetcd.EtcdException`

Error ecodeNameRequired (http code 206)

exception `pyetcd.EtcdNoMorePeer`

Bases: `pyetcd.EtcdException`

Error ecodeNoMorePeer (http code 103)

exception `pyetcd.EtcdNodeExist`

Bases: `pyetcd.EtcdException`

Error EcodeNodeExist (http code 105)

exception `pyetcd.EtcdNotDir`

Bases: `pyetcd.EtcdException`

Error EcodeNotDir (http code 104)

exception `pyetcd.EtcdNotFile`

Bases: `pyetcd.EtcdException`

Error EcodeNotFile (http code 102)

exception `pyetcd.EtcdPrevValueRequired`

Bases: `pyetcd.EtcdException`

Error EcodePrevValueRequired (http code 201)

exception `pyetcd.EtcdRaftInternal`

Bases: `pyetcd.EtcdException`

Error EcodeRaftInternal (http code 300)

exception `pyetcd.EtcdRefreshTTLRequired`

Bases: `pyetcd.EtcdException`

Error EcodeRefreshTTLRequired (http code 212)

exception `pyetcd.EtcdRefreshValue`

Bases: `pyetcd.EtcdException`

Error EcodeRefreshValue (http code 211)

class `pyetcd.EtcdResult` (*response*)

Bases: `object`

Response from Etcd API.

Parameters `response` (`requests.Response`) – Response from server as `requests`.
(`get` | `post` | `put`) returns.

Raises

- `EtcdException` – if response contains non-200 `errorCode`.
- `EtcdInvalidResponse` – if payload is invalid.
- `EtcdEmptyResponse` – if response content from etcd is empty.

action
Action type

followers
Followers of leader

health
name

id
etcd identifier of the etcd node.

leader
Leader of cluster

leaderInfo

name
etcd name of the node.

node
Node class instance. It holds the current key value.

prevNode
Node class instance. It holds the previous key value.

recvAppendRequestCnt

sendAppendRequestCnt

startTime

state

version_etcdcluster
Version of Etcd cluster

version_etcdserver
Version of Etcd server

x_etcd_index
current etcd index that represents key modification version.

exception `pyetcd.EtcdRootOnly`
Bases: `pyetcd.EtcdException`
Error EcodeRootOnly (http code 107)

exception `pyetcd.EtcdStandbyInternal`
Bases: `pyetcd.EtcdException`
Error ecodeStandbyInternal (http code 402)

exception `pyetcd.EtcdTTLNaN`
Bases: `pyetcd.EtcdException`
Error EcodeTTLNaN (http code 202)

exception `pyetcd.EtcdTestFailed`
Bases: `pyetcd.EtcdException`
Error EcodeTestFailed (http code 101)

exception `pyetcd.EtcdTimeoutNaN`
Bases: `pyetcd.EtcdException`

Error `TimeoutNaN` (http code 205)

exception `pyetcd.EtcdUnauthorized`

Bases: `pyetcd.EtcdException`

Error `EcodeUnauthorized` (http code 110)

exception `pyetcd.EtcdValueOrTTLRequired`

Bases: `pyetcd.EtcdException`

Error `ecodeValueOrTTLRequired` (http code 204)

exception `pyetcd.EtcdValueRequired`

Bases: `pyetcd.EtcdException`

Error `ecodeValueRequired` (http code 200)

exception `pyetcd.EtcdWatcherCleared`

Bases: `pyetcd.EtcdException`

Error `EcodeWatcherCleared` (http code 400)

Contributions are welcome, and they are greatly appreciated! Every little bit helps, and credit will always be given. You can contribute in many ways:

5.1 Types of Contributions

5.1.1 Report Bugs

Report bugs at <https://github.com/twindb/pyetcd/issues>.

If you are reporting a bug, please include:

- Your operating system name and version.
- Any details about your local setup that might be helpful in troubleshooting.
- Detailed steps to reproduce the bug.

5.1.2 Fix Bugs

Look through the GitHub issues for bugs. Anything tagged with “bug” and “help wanted” is open to whoever wants to implement it.

5.1.3 Implement Features

Look through the GitHub issues for features. Anything tagged with “enhancement” and “help wanted” is open to whoever wants to implement it.

5.1.4 Write Documentation

pyetcd could always use more documentation, whether as part of the official pyetcd docs, in docstrings, or even on the web in blog posts, articles, and such.

5.1.5 Submit Feedback

The best way to send feedback is to file an issue at <https://github.com/twindb/pyetcd/issues>.

If you are proposing a feature:

- Explain in detail how it would work.
- Keep the scope as narrow as possible, to make it easier to implement.
- Remember that this is a volunteer-driven project, and that contributions are welcome :)

5.2 Get Started!

Ready to contribute? Here's how to set up *pyetcd* for local development.

1. Fork the *pyetcd* repo on GitHub.
2. Clone your fork locally:

```
$ git clone git@github.com:your_name_here/pyetcd.git
```

3. Install your local copy into a virtualenv. Assuming you have virtualenvwrapper installed, this is how you set up your fork for local development:

```
$ mkvirtualenv pyetcd
$ cd pyetcd/
$ python setup.py develop
```

4. Create a branch for local development:

```
$ git checkout -b name-of-your-bugfix-or-feature
```

Now you can make your changes locally.

5. When you're done making changes, check that your changes pass flake8 and the tests, including testing other Python versions with tox:

```
$ flake8 pyetcd tests
$ python setup.py test or py.test
$ tox
```

To get flake8 and tox, just pip install them into your virtualenv.

6. Commit your changes and push your branch to GitHub:

```
$ git add .
$ git commit -m "Your detailed description of your changes."
$ git push origin name-of-your-bugfix-or-feature
```

7. Submit a pull request through the GitHub website.

5.3 Pull Request Guidelines

Before you submit a pull request, check that it meets these guidelines:

1. The pull request should include tests.
2. If the pull request adds functionality, the docs should be updated. Put your new functionality into a function with a docstring, and add the feature to the list in README.rst.
3. The pull request should work for Python 2.6, 2.7, 3.4 and 3.5, and for PyPy. Check https://travis-ci.org/twindb/pyetcd/pull_requests and make sure that the tests pass for all supported Python versions.

5.4 Tips

To run a subset of tests:

```
$ py.test tests/test_client.py
$ py.test tests/test_etcdresult.py
```

To run individual tests:

```
$ py.test tests/test_client.py::test_write
```


CHAPTER 6

Credits

6.1 Development Lead

- TwinDB Development Team <dev@twindb.com>

6.2 Contributors

None yet. Why not be the first?

7.1 0.1.0 (2016-09-17)

- First release on PyPI.

CHAPTER 8

Indices and tables

- `genindex`
- `modindex`
- `search`

p

`pyetcd`, [12](#)

`pyetcd.client`, [9](#)

A

action (pyetcd.EtcdResult attribute), 14
add_member() (pyetcd.client.Client method), 9

C

Client (class in pyetcd.client), 9
ClientException, 12
compare_and_delete() (pyetcd.client.Client method), 10
compare_and_swap() (pyetcd.client.Client method), 10

D

delete() (pyetcd.client.Client method), 10

E

EtcdClientInternal, 12
EtcdDirNotEmpty, 12
EtcdEmptyResponse, 12
EtcdEventIndexCleared, 12
EtcdException, 13
EtcdExistingPeerAddr, 13
EtcdIndexNaN, 13
EtcdIndexOrValueRequired, 13
EtcdIndexValueMutex, 13
EtcdInvalidActiveSize, 13
EtcdInvalidField, 13
EtcdInvalidForm, 13
EtcdInvalidRemoveDelay, 13
EtcdInvalidResponse, 13
EtcdKeyIsPreserved, 13
EtcdKeyNotFound, 13
EtcdLeaderElect, 13
EtcdNameRequired, 14
EtcdNodeExist, 14
EtcdNoMorePeer, 14
EtcdNotDir, 14
EtcdNotFile, 14
EtcdPrevValueRequired, 14
EtcdRaftInternal, 14
EtcdRefreshTTLRequired, 14

EtcdRefreshValue, 14
EtcdResult (class in pyetcd), 14
EtcdRootOnly, 15
EtcdStandbyInternal, 15
EtcdTestFailed, 15
EtcdTimeoutNaN, 15
EtcdTTLNaN, 15
EtcdUnauthorized, 16
EtcdValueOrTTLRequired, 16
EtcdValueRequired, 16
EtcdWatcherCleared, 16

F

followers (pyetcd.EtcdResult attribute), 15

H

health (pyetcd.client.Client attribute), 11
health (pyetcd.EtcdResult attribute), 15

I

id (pyetcd.EtcdResult attribute), 15

L

leader (pyetcd.EtcdResult attribute), 15
leaderInfo (pyetcd.EtcdResult attribute), 15

M

mkdir() (pyetcd.client.Client method), 11

N

name (pyetcd.EtcdResult attribute), 15
node (pyetcd.EtcdResult attribute), 15

P

prevNode (pyetcd.EtcdResult attribute), 15
pyetcd (module), 12
pyetcd.client (module), 9

R

`read()` (`pyetcd.client.Client` method), [11](#)
`recvAppendRequestCnt` (`pyetcd.EtcdResult` attribute), [15](#)
`remove_member()` (`pyetcd.client.Client` method), [11](#)
`rmdir()` (`pyetcd.client.Client` method), [11](#)

S

`sendAppendRequestCnt` (`pyetcd.EtcdResult` attribute), [15](#)
`startTime` (`pyetcd.EtcdResult` attribute), [15](#)
`state` (`pyetcd.EtcdResult` attribute), [15](#)

U

`update_ttl()` (`pyetcd.client.Client` method), [11](#)

V

`version()` (`pyetcd.client.Client` method), [11](#)
`version_cluster()` (`pyetcd.client.Client` method), [12](#)
`version_etcdcluster` (`pyetcd.EtcdResult` attribute), [15](#)
`version_etcdserver` (`pyetcd.EtcdResult` attribute), [15](#)
`version_server()` (`pyetcd.client.Client` method), [12](#)

W

`write()` (`pyetcd.client.Client` method), [12](#)

X

`x_etcd_index` (`pyetcd.EtcdResult` attribute), [15](#)