pyethapp Documentation

Release 0.9.16-17a5-dirty

HeikoHeiko

Nov 09, 2017

Contents

1 pyethapp **3**
- 1.1 Introduction . . . 3
- 1.2 Installation and invocation . . . 3
- 1.3 Status . . . 3

2 Installation **5**

3 Usage **7**

4 Contributing **9**
- 4.1 Types of Contributions . . . 9
- 4.2 Get Started! . . . 10
- 4.3 Pull Request Guidelines . . . 11
- 4.4 Tips . . . 11

5 Credits **13**
- 5.1 Development Lead . . . 13
- 5.2 Contributors . . . 13

6 History **15**

7 0.1.0 (20150-1-1) **17**

8 Indices and tables **19**

Contents:

CHAPTER 1

pyethapp

1.1 Introduction

pyethapp is the python based client implementing the Ethereum cryptoeconomic state machine.

Ethereum as a platform is focussed on enabling people to build new ideas using blockchain technology.

The python implementation aims to provide an easily hackable and extendable codebase.

pyethapp leverages two ethereum core components to implement the client:

- pyethereum - the core library, featuring the blockchain, the ethereum virtual machine, mining
- pydevp2p - the p2p networking library, featuring node discovery for and transport of multiple services over multiplexed and encrypted connections

1.2 Installation and invocation

- git clone https://github.com/ethereum/pyethapp
- cd pyethapp
- python setup.py install
- pyethapp (shows help)
- pyethapp run (starts the client)

There is also Dockerfile in the repo.

1.3 Status

- Working PoC9 prototype

- interoperable with the go and cpp clients
- jsonrpc (mostly)

CHAPTER 2

Installation

At the command line:

```
$ easy_install pyethapp
```

Or, if you have virtualenvwrapper installed:

```
$ mkvirtualenv pyethapp
$ pip install pyethapp
```

CHAPTER 3

Usage

To use pyethapp in a project:

```
import pyethapp
```

CHAPTER 4

Contributing

Contributions are welcome, and they are greatly appreciated! Every little bit helps, and credit will always be given.

You can contribute in many ways:

4.1 Types of Contributions

4.1.1 Report Bugs

Report bugs at https://github.com/ethereum/pyethapp/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.

4.1.2 Fix Bugs

Look through the GitHub issues for bugs. Anything tagged with "bug" is open to whoever wants to implement it.

4.1.3 Implement Features

Look through the GitHub issues for features. Anything tagged with "feature" is open to whoever wants to implement it.

4.1.4 Write Documentation

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

4.1.5 Submit Feedback

The best way to send feedback is to file an issue at https://github.com/ethereum/pyethapp/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 :)

4.2 Get Started!

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

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

```
$ git clone git@github.com:your_name_here/pyethapp.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 pyethapp
$ cd pyethapp/
$ 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 pyethapp tests
$ python setup.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.

4.3 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, 3.3, and 3.4, and for PyPy. Check https://travis-ci.org/ethereum/pyethapp/pull_requests and make sure that the tests pass for all supported Python versions.

4.4 Tips

To run a subset of tests:

```
$ python -m unittest tests.test_pyethapp
```

CHAPTER 5

Credits

5.1 Development Lead

- HeikoHeiko <heiko@ethdev.com>

5.2 Contributors

None yet. Why not be the first?

CHAPTER 6

History

CHAPTER 7

0.1.0 (20150-1-1)

- First release on PyPI.

CHAPTER 8

Indices and tables

- genindex
- modindex
- search