MBot Documentation

Release 0.0.2

Michael Kutý

February 13, 2017

Contents

Contents:

CHAPTER 1

MBot

Yet another Bot implemented in pure Python. Is designed to support multiple backends with global state machine which can store shortcuts for your commands and more. Support user permissions for actions.

- Free software: MIT license
- Documentation: https://mbot.readthedocs.io.

1.1 Message Backends

- Slack
- MQTT (in dev)
- RabbitMQ (TODO)

1.2 Storage Backends

- Local file
- S3 file
- Support encryption in default
- DB (TODO)

1.3 Middlewares

- Slack
- SQL
- AirflowTrigger
- Bash
- Saltstack (pepper client)
- Joker (say some joke)
- History (list of your commands, save shortcuts)

- Help
- Jinja2 Template

1.4 Features

- Simple scheduler
- User roles and permissions
- Filebased config/state
- Encrypted storage
- Connections management
- S3 storage
- Dialogs
- Auto installing of dependencies
- Dynamic loading
- Render messages with Jinja2

1.5 Installation

```
pip install mbot
```

1.6 Usage

```
$ mbot run
Slack token: xoxb-46465446310727-654564564564654565456
Starting your BOT...
INFO:requests.packages.urllib3.connectionpool:Starting new HTTPS connection (1): slack.com
```

Config

```
$ mbot config
{'core': {'backends': ['slack'],
          'config_path': '/Users/user/Library/Application '
                         'Support/mbot/mbot.conf',
          'data_path': '/Users/user/Library/Application '
                       'Support/mbot/state.db'},
 'logging': {'verbosity': 'INFO'},
 'slack': {'engine': 'mbot.backends.slack.Slack',
           'middlewares': ['mbot.contrib.system.Config',
                           'mbot.contrib.history.history',
                           'mbot.contrib.debug.Debug',
                           'mbot.contrib.bash.Bash',
                           'mbot.contrib.console.MBotConsole',
                           'mbot.contrib.python.Python',
                           'mbot.contrib.joker.Joker',
                           'mbot.contrib.hackernews.HackerNews',
                           'mbot.contrib.scheduler.Scheduler',
```

```
                            'mbot.contrib.salt.Salt',
                            'mbot.contrib.connections.Connections',
                            'mbot.contrib.dialogs.Dialogs',
                            'mbot.contrib.airflow.AirflowTrigger',
                            'mbot.contrib.sql.SQL',
                            'mbot.contrib.help.Help'],
          'token': 'xoxb-46465446310727-654564564564654565456'},
 'storage': {'encrypt': True,
             'engine': 'local',
             'fernet_token': 'oMdNGsFou566j4e3SL6cij3HR70D-xIqh58z30B2BAs='}}
```

Add user to admin group

```
mbot: users.all()
```

```
mbot: users.update("your_user_id", ["admin"], "groups")
```

1.7 TODO

- Variables, management
- Support Celery as executor
- SSH
- Use appdirs when data-path is not provied
- Chain of commands

CHAPTER 2

Installation

2.1 Stable release

To install MBot, run this command in your terminal:

```
$ pip install mbot
```

This is the preferred method to install MBot, 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 Optional dependencies

```
$ pip install uvloop
```

2.3 From sources

The sources for MBot can be downloaded from the Github repo.

You can either clone the public repository:

```
$ git clone git://github.com/michaelkuty/mbot
```

Or download the tarball:

```
$ curl  -OL https://github.com/michaelkuty/mbot/tarball/master
```

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

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

CHAPTER 3

Usage

To use MBot in a project:

```
$ mbot run
Slack token: xoxb-46465446310727-654564564564654565456
Starting your BOT...
INFO:requests.packages.urllib3.connectionpool:Starting new HTTPS connection (1): slack.com
```

Config

```
$ mbot config
{'core': {'backends': ['slack'],
          'config_path': '/Users/user/Library/Application '
                         'Support/mbot/mbot.conf',
          'data_path': '/Users/user/Library/Application '
                       'Support/mbot/state.db'},
 'logging': {'verbosity': 'INFO'},
 'slack': {'engine': 'mbot.backends.slack.Slack',
           'middlewares': ['mbot.contrib.system.Config',
                           'mbot.contrib.history.history',
                           'mbot.contrib.debug.Debug',
                           'mbot.contrib.bash.Bash',
                           'mbot.contrib.console.MBotConsole',
                           'mbot.contrib.python.Python',
                           'mbot.contrib.joker.Joker',
                           'mbot.contrib.hackernews.HackerNews',
                           'mbot.contrib.scheduler.Scheduler',
                           'mbot.contrib.salt.Salt',
                           'mbot.contrib.connections.Connections',
                           'mbot.contrib.dialogs.Dialogs',
                           'mbot.contrib.airflow.AirflowTrigger',
                           'mbot.contrib.sql.SQL',
                           'mbot.contrib.help.Help'],
           'token': 'xoxb-46465446310727-654564564564654565456'},
 'storage': {'encrypt': True,
             'engine': 'local',
             'fernet_token': 'oMdNGsFou566j4e3SL6cij3HR70D-xIqh58z30B2BAs='}}
```

Add user to admin group

```
mbot: users.all()
```

```
mbot: users.update("your_user_id", ["admin"], "groups")
```

3.1 Storing commands

```
save my last command name: test
save my command index: 0 name: test
```

List of commands

```
list
```

3.2 Help

```
help history
help salt
```

3.3 Scheduled Jobs

```
mbot: jobs.all()
```

delete

```
mbot: jobs.delete("name")
```

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/michaelkuty/sbot/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" and "help wanted" is open to whoever wants to implement it.

4.1.3 Implement Features

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

4.1.4 Write Documentation

Slack Bot could always use more documentation, whether as part of the official Slack Bot 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/michaelkuty/sbot/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 *sbot* for local development.

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

```
$ git clone git@github.com:your_name_here/sbot.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 sbot
$ cd sbot/
$ 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 sbot 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.

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, 3.4 and 3.5, and for PyPy. Check https://travis-ci.org/michaelkuty/sbot/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_sbot
```

CHAPTER 5

Indices and tables

- genindex
- modindex
- search