supervisor-sysinfo

Release 0.1.0

February 21, 2016

Contents

CHAPTER 1

Overview

a supervisor rpc extension for obtaining the hosts system and process information plus a supervisor event monitor to collect system stats and post them to a REST endpoint.

- Free software: BSD license

1.1 Installation

```
pip install git+https://github.com/thanos/supervisor-sysinfo.git
```

1.2 Documentation

https://supervisor-sysinfo.readthedocs.org/

CHAPTER 2

supervisor-sysinfo

a supervisor rpc extension for obtaining the hosts system and process information.

2.1 Requirements

- Python 2.7+
- supervisor 3.0+
- psutil

2.2 Why

I use supervisor on many of my servers for control and monitoring and I needed a way to interrogate the sevrers for top, df and

- sysinfo.ps which returns a dictionary keyed by pid of the output of ps aux.
- sysinfo.sysInfo which reruns cpu, memory and disk usage information.

Both functions return json strings. I do this as a work around the 32-bit int limitation of the XML-RPC standard.

2.3 Installation

Use the usual ways:

```
pip install git+https://github.com/thanos/supervisor-sysinfo.git
```

2.4 Setup

In your supervisor.conf file setup:

```
[inet_http_server]
port = *:9002
username = very_safe
password = very_safe
```

Uncomment:

```
[rpcinterface:supervisor]
supervisor.rpcinterface_factory = supervisor.rpcinterface:make_main_rpcinterface
```

Add:

```
      [rpcinterface:sysinfo]
      supervisor.rpcinterface_factory = supervisor_sysinfo.rpcinterface:make_sysinfo_rpcinterface

[eventlistener:monitor]
command=supervisor_monitor http://some_end_point.com:7000/monitor
events= TICK_60
```

2.5 Usage

You can look at the test code but effectively you need to do this:

```
import xmlrpclib,pprint, json

rpc_proxy =  xmlrpclib.ServerProxy('http://very_safe:very_safe@127.0.0.1:9002')

ps_list = json.loads(rpc_proxy.sysinfo.ps())
pprint.pprint(ps_list)

sysInfo =  json.loads(rpc_proxy.sysinfo.sysInfo())
pprint.pprint(sysInfo)
```

2.6 Development

To run the all tests run:

```
tox
```

CHAPTER 3

Installation

At the command line:

```
pip install supervisor_sysinfo
```

CHAPTER 4

Usage

To use supervisor-sysinfo in a project:

```
import supervisor_sysinfo
```

CHAPTER 5

Reference

5.1 supervisor_sysinfo

CHAPTER 6

Contributing

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

6.1 Bug reports

When 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.

6.2 Documentation improvements

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

6.3 Feature requests and feedback

The best way to send feedback is to file an issue at https://github.com/thanos/supervisor-sysinfo/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 code contributions are welcome :)

6.4 Development

To set up *supervisor-sysinfo* for local development:

1. Fork supervisor-sysinfo on GitHub.
2. Clone your fork locally:

```
git clone git@github.com:your_name_here/supervisor-sysinfo.git
```

3. Create a branch for local development:

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

   Now you can make your changes locally.

4. When you're done making changes, run all the checks, doc builder and spell checker with tox one command:

```
tox
```

5. 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
```

6. Submit a pull request through the GitHub website.

6.4.1 Pull Request Guidelines

If you need some code review or feedback while you're developing the code just make the pull request.

For merging, you should:

1. Include passing tests (run `tox`) [1].
2. Update documentation when there's new API, functionality etc.
3. Add a note to `CHANGELOG.rst` about the changes.
4. Add yourself to `AUTHORS.rst`.

6.4.2 Tips

To run a subset of tests:

```
tox -e envname -- py.test -k test_myfeature
```

To run all the test environments in *parallel* (you need to `pip install detox`):

```
detox
```

[1] If you don't have all the necessary python versions available locally you can rely on Travis - it will run the tests for each change you add in the pull request.
It will be slower though ...

CHAPTER 7

Authors

- Thanos Vassilakis - https://github.com/thanos/thanos

CHAPTER 8

Changelog

8.1 0.1.0 (2016-01-19)

- First release on PyPI.

CHAPTER 9

Indices and tables

- genindex
- modindex
- search

S

S