
PythonEM7 Documentation

Release 0.5.0

Anthony Shaw

Jul 10, 2018

Contents

1	ScienceLogic - a Python library for the ScienceLogic EM7 API	3
1.1	Usage	3
1.2	Credits	3
2	Installation	5
3	Usage	7
4	API	9
4.1	Client API	9
4.2	Device API	9
4.3	Performance Data API	10
4.4	Presentation API	10
5	Contributing	13
5.1	Types of Contributions	13
5.2	Get Started!	14
5.3	Pull Request Guidelines	15
5.4	Tips	15
6	Credits	17
6.1	Development Lead	17
6.2	Contributors	17
7	History	19
7.1	0.5.0 (2016-07-12)	19
7.2	0.1.0 (2016-04-04)	19
8	Indices and tables	21

Contents:

ScienceLogic - a Python library for the ScienceLogic EM7 API

Client library for sciencelogic EM7

- Free software: MIT license
- Documentation: <https://sciencelogic.readthedocs.org>.

1.1 Usage

To use Python EM7 in a project

```
from sciencelogic.client import Client

c = Client('jazz', 'hands!', 'https://au-monitoring.mcp-services.net/')

# API details
print(c.sysinfo)
```

1.2 Credits

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

Thanks to Georgi Dimitrov (@georgijd) for his contributions and testing

CHAPTER 2

Installation

At the command line:

```
$ easy_install sciencellogic
```

Or, if you have virtualenvwrapper installed:

```
$ mkvirtualenv sciencellogic  
$ pip install sciencellogic
```


CHAPTER 3

Usage

To use PythonEM7 in a project:

```
from sciencelogic.client import Client

c = Client('jazz', 'hands!', 'https://au-monitoring.mcp-services.net/')

# API details
print(c.sysinfo)
```


4.1 Client API

```
class sciencelogic.client.Client (username, password, uri, auto_connect=True, verify_ssl=False)
```

```
    devices (details=False, limit=100)  
        Get a list of devices
```

Parameters

- **details** (*int*) – Get the details of the devices
- **limit** – Number of devices to retrieve

Return type *list of Device*

```
    get (uri, params=None)  
        Get a URI from the API
```

Parameters **uri** (*str*) – The URI

Params **params** Extra params

```
    get_device (device_id)  
        Get a devices
```

Parameters **device_id** (*int*) – The id of the device

Return type *list of Device*

4.2 Device API

```
class sciencelogic.device.Device (device, uri, client, has_details=False, fetch_details=False)  
    Represents a monitored device
```

get_logs (*extended_fetch=0, hide_filter_info=1, link_disp_field=None, limit=1000, offset=None*)

Get logs for this device

Parameters

- **extended_fetch** (*bool*) – Fetch entire resource if 1 (true), or resource link only if 0 (false).
- **hide_filter_info** (*bool*) – Suppress filterspec and current filter info
- **link_disp_field** (*list*) – When not using `extended_fetch`, this determines which field is used for the “description” of the resource link
- **limit** (*int*) – Number of records to retrieve
- **offset** (*int*) – Specifies the index of the first returned resource within the entire result set

Return type *list of dict*

performance_counters ()

Get a list of performance counters for this device

Return type *list of PerformanceData*

4.3 Performance Data API

class `sciencelogic.performance_data.PerformanceData` (*client, options*)

get_presentation_data (*presentation*)

Get the data for a presentation object

Parameters **presentation** (*Presentation*) – The presentation of this counter

Return type *dict*

get_presentations ()

Get a list of presentations for this performance counter

Return type *list of Presentation*

4.4 Presentation API

class `sciencelogic.presentations.Presentation` (*presentation_dict, client*)

A presentation of a monitor’s data

get_data (*beginstamp=None, endstamp=None, insert_nulls=1, fetch_indexes=0, duration=None, idx_list=None, idx_max=None, hide_options=1*)

Get the data for this presentation

Parameters

- **beginstamp** (*str*) – timestamp for the beginning of the desired range of data
- **endstamp** (*str*) – timestamp for the end of the desired range of data
- **insert_nulls** (*bool*) – insert NULLs for all poll dates within the specified date range that do not have polled values

- **fetch_indexes** (`bool`) – fetch the list of collection indexes (and their string labels, if any exist) for the date range instead of actual data.
- **duration** (`str`) – human readable short-hand, such as 24h, 5d, 90m where h=HOUR, d=DAY, m=MINUTE. Used to specify the amount of data to fetch
- **idx_list** (`list`) – a list of collection indexes for which to fetch data (when not specified, all indexes up to `idx.max` will be fetched)
- **idx_max** (`int`) – limits the maximum number of collection indexes to return for the requested presentation data
- **hide_options** (`bool`) – hide the available request parameters from the response

Return type `dict`

Note: if `fetch_indexes` is enabled, `idx.max` and `idx.list` will be ignored.

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/tonybaloney/scienceologic/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” is open to whoever wants to implement it.

5.1.3 Implement Features

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

5.1.4 Write Documentation

PythonEM7 could always use more documentation, whether as part of the official PythonEM7 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/tonybaloney/sciencelogic/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 *sciencelogic* for local development.

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

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

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, 3.3, 3.4 and 3.5, and for PyPy. Check https://travis-ci.org/tonybaloney/sciencellogic/pull_requests and make sure that the tests pass for all supported Python versions.

5.4 Tips

To run a subset of tests:

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


6.1 Development Lead

- Anthony Shaw <anthonyshaw@apache.org>

6.2 Contributors

None yet. Why not be the first?

7.1 0.5.0 (2016-07-12)

- Add filtering options for the *get_data* method

7.2 0.1.0 (2016-04-04)

- First release on PyPI.

CHAPTER 8

Indices and tables

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

C

Client (class in `sciencelogic.client`), 9

D

Device (class in `sciencelogic.device`), 9

`devices()` (`sciencelogic.client.Client` method), 9

G

`get()` (`sciencelogic.client.Client` method), 9

`get_data()` (`sciencelogic.presentations.Presentation` method), 10

`get_device()` (`sciencelogic.client.Client` method), 9

`get_logs()` (`sciencelogic.device.Device` method), 9

`get_presentation_data()` (`sciencelogic.performance_data.PerformanceData` method), 10

`get_presentations()` (`sciencelogic.performance_data.PerformanceData` method), 10

P

`performance_counters()` (`sciencelogic.device.Device` method), 10

`PerformanceData` (class in `sciencelogic.performance_data`), 10

`Presentation` (class in `sciencelogic.presentations`), 10