sugar Documentation

Release 0.1.1

Bharadwaj Yarlagadda

October 11, 2016

Contents

Python utility library. Based on sugar Javascript Library.

CHAPTER 1

Links

- Project: https://github.com/bharadwajyarlagadda/sugar.py
- Documentation: http://sugarpy.readthedocs.io
- Pypi: https://pypi.python.org/pypi/sugar.py
- TravisCI: https://travis-ci.org/bharadwajyarlagadda/sugar.py

CHAPTER 2

Features

- Supported on Python 2.7 and Python 3.3+.

CHAPTER 3

Quickstart

Install using pip:

```
pip install sugar.py
```

```
>>> import sugar

>>> float(sugar.average([1, 2, 3]))
2.0

>>> sugar.construct(4, lambda x: x * 2)
[0, 2, 4, 6]

>>> sugar.count([1, 2, 3, 3], 3)
2

>>> sugar.subtract([1, 2, 3], 2)
[1, 3]
>>> sugar.subtract ([1, 2, 3], [1, 3])
[2]
>>> sugar.subtract([1, 2, 3], 4)
[1, 2, 3]
```

CHAPTER 4

Guide

4.1 Installation

sugar.py requires Python 2.7 or 3.3+.

To install from PyPI:

```
pip install sugar.py
```

You can also install sugar.py with all the latest changes:

```
$ git clone git@github.com:bharadwajyarlagadda/sugar.py.git
$ cd sugar.py
$ python setup.py install
```

4.2 API Reference

`sugar.arrays.`**`average`**(*array*)

Returns the average for all the values in the given `array`.

Parameters **`array`** (*`list`*) – List of values.

Returns Average of all the values in the given list.

Return type int/float

Example

```
>>> float(average([1, 2, 3]))
2.0
```

New in version 0.1.0.

`sugar.arrays.`**`construct`**(*var*, *callback*)

Constructs an array of `var` length from the values of `callback`.

Parameters

- **`var`** (*`int`*) – Length of the array intended.
- **`callback`** – A method that can take in each variable from the given range and return back a new value based on the method definition.

Returns A list of `callback` values.

Return type list

Example

```
>>> construct(4, lambda x: x * 2)
[0, 2, 4, 6]
```

New in version 0.1.0.

`sugar.arrays.`**`count`**(*array*, *value*)

Counts all elements in the `array` that match the given `value`.

Parameters

- **`array`** (*`list`*) – A list of values provided by the user to search for.
- **`value`** (*`int/float/str`*) – Value that needs to be counted.

Returns Count of the given value.

Return type int

Example

```
>>> count([1, 2, 3, 3], 3)
2
```

New in version 0.1.0.

`sugar.arrays.`**`subtract`**(*array*, *item*)

Subtracts `item` from the `array` and returns the result as a new array. If `item` is also an array, all elements in it will be removed.

Parameters

- **`array`** (*`list`*) – A list of values provided by the user.
- **`item`** (*`list/int/float/str`*) – A value that needs to be removed from `array`.

Returns A new list with the `item` removed.

Return type list

Example

```
>>> subtract([1, 2, 3], 2)
[1, 3]
>>> subtract ([1, 2, 3], [1, 3])
[2]
>>> subtract([1, 2, 3], 4)
[1, 2, 3]
```

New in version 0.1.0.

CHAPTER 5

Project Info

5.1 License

5.2 Versioning

This project follows Semantic Versioning.

It is recommended to only use or import objects from the main module, sugar.py.

5.3 Changelog

5.3.1 v0.1.1 (2016-10-10)

- `FIX`: Added description for the package in `setup.py`.
- `FIX`: Added keywords in `setup.py`.

5.3.2 v0.1.0 (2016-10-10)

- First release.
- Add `average()`.
- Add `construct()`.
- Add `count()`.
- Add `subtract()`.

5.4 Authors

5.4.1 Lead

- Bharadwaj Yarlagadda, yarlagaddabharadwaj@gmail.com, bharadwajyarlagadda@github

5.5 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:

5.5.1 Types of Contributions

Report Bugs

Report bugs at https://github.com/bharadwajyarlagadda/sugar.py.

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.

Fix Bugs

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

Implement Features

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

Write Documentation

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

Submit Feedback

The best way to send feedback is to file an issue at https://github.com/bharadwajyarlagadda/sugar.py.

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 :)

5.5.2 Get Started!

Ready to contribute? Here's how to set up `sugar.py` for local development.

1. Fork the `sugar.py` repo on GitHub.
2. Pull your fork locally:

```
$ git clone git@github.com:<username>/sugar.py.git
```

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

```
$ cd sugar.py/
$ pip install -r requirements-dev.txt
```

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 linting and all unit tests by testing with tox across all supported Python versions:

```
$ invoke tox
```

6. Add yourself to `AUTHORS.rst`.
7. Commit your changes and push your branch to GitHub:

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

8. Submit a pull request through the GitHub website.

5.5.3 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 README.rst.
3. The pull request should work for Python 2.7, 3.4, and 3.5. Check https://travis-ci.org/bharadwajyarlagadda/sugar.py/pull_requests and make sure that the tests pass for all supported Python versions.

CHAPTER 6

Indices and tables

- genindex
- modindex
- search

Index