fuzzyfinder Documentation

Release 2.1.0

Amjith Ramanujam

Aug 27, 2018

Contents

Contents:

CHAPTER 1

fuzzyfinder

Fuzzy Finder implemented in Python. Matches partial string entries from a list of strings. Works similar to fuzzy finder in SublimeText and Vim's Ctrl-P plugin.

- Documentation: https://fuzzyfinder.readthedocs.org.
- Source: https://github.com/amjith/fuzzyfinder

1.1 Quick Start

```
$ pip install fuzzyfinder

or

$ easy_install fuzzyfinder
```

1.2 Usage

```
>>> from fuzzyfinder import fuzzyfinder

>>> suggestions = fuzzyfinder('abc', ['defabca', 'abcd', 'aagbec', 'xyz', 'qux'])
>>> list(suggestions)
['abcd', 'defabca', 'aagbec']

>>> # Use a user-defined function to obtain the string against which fuzzy matching␣
↪is done
>>> collection = ['aa bbb', 'aca xyz', 'qx ala', 'xza az', 'bc aa', 'xy abca']
>>> suggestions = fuzzyfinder('aa', collection, accessor=lambda x: x.split()[1])
>>> list(suggestions)
```

(continues on next page)

(continued from previous page)

```
['bc aa', 'qx ala', 'xy abca']

>>> suggestions = fuzzyfinder('aa', ['aac', 'aaa', 'aab', 'xyz', 'ada'])
>>> list(suggestions)
['aaa', 'aab', 'aac', 'ada']

>>> # Preserve original order of elements if matches have same rank
>>> suggestions = fuzzyfinder('aa', ['aac', 'aaa', 'aab', 'xyz', 'ada'], sort_
↪results=False)
>>> list(suggestions)
['aac', 'aaa', 'aab', 'ada']
```

1.3 Features

- Simple, easy to understand code.
- No external dependencies, just the python std lib.

1.4 How does it work

Blog post describing the algorithm: http://blog.amjith.com/fuzzyfinder-in-10-lines-of-python

1.5 Similar Projects

- https://github.com/seatgeek/fuzzywuzzy - Fuzzy matching and auto-correction using levenshtein distance.

CHAPTER 2

Installation

At the command line:

```
$ pip install fuzzyfinder
```

Or:

```
$ easy_install fuzzyfinder
```

Or, if you have virtualenvwrapper installed:

```
$ mkvirtualenv fuzzyfinder
$ pip install fuzzyfinder
```

CHAPTER 3

Usage

To use fuzzyfinder in a project:

```
from fuzzyfinder import fuzzyfinder

collection = ['abcd', 'defabca', 'aagbec', 'xyz', 'qux']

suggestions = fuzzyfinder('abc', collection)

print list(suggestions)
```

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/amjith/fuzzyfinder/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

fuzzyfinder could always use more documentation, whether as part of the official fuzzyfinder 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/amjith/fuzzyfinder/issues.

4.2 Get Started!

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

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

```
$ git clone git@github.com:your_name_here/fuzzyfinder.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 fuzzyfinder
$ cd fuzzyfinder/
$ pip install -r dev-requirements.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 the tests. This project uses py.test for testing and tox for running the tests on multiple versions of python:

```
$ py.test
$ tox
```

   To get tox, just pip install tox 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.

CHAPTER 5

Credits

5.1 Project Lead

- Amjith Ramanujam

5.2 Contributors

- Gokul Soumya
- Luke Murphy
- Matheus Rosa

CHAPTER 6

Changelog

CHAPTER 7

2.0.0 (2017-01-25)

- Case insensitive matching. (Gokul Soumya)
- Add an accessor function for fuzzy find. (Amjith)
- Support integer inputs. (Matheus)

CHAPTER 8

2.1.0 (2017-01-25)

- Use lookahead regex to find shortest match. (Gokul Soumya)

CHAPTER 9

Indices and tables

- genindex
- modindex
- search