random_profile

Release 3.0.2

Deepak Raj

Nov 08, 2023

CONTENTS

RandomProfileGenerator is a powerful and simple tool to generate fake data. You can use it to mock classes, populate databases and much more. You can check the full documentation here.

RandomProfileGenerator project developed under the MIT license by Py-Contributors. Py-Contributors is open source community of developers who are working on various python projects.

If you love open source contributions.

- Join the community on Discord.
- join the community on Github.

CHAPTER

ONE

OTHER PROJECTS BY PY-CONTRIBUTORS:

- AudioBook
- Cybel - The Discord Bot
- Twitterify - Tweet-retweet bot

1.1 Introduction

RandomProfileGenerator is a powerful and simple tool to generate fake data. You can use it to mock classes, populate databases and much more. You can check the full documentation here. It is written in python and is compatible with python 3.7+. It is also compatible with Windows, Linux and Mac OS X.

Use cases

- Mocking classes
- Populating databases
- Generating test data
- Generating fake data for Cybersecurity
- Generating fake data for your tests
- Generating fake data for your documentation
- Generating fake data for your presentation
- Generating fake data for your demo
- Generating fake data for your blog post

1.1.1 Upcoming features

- Support for more languages (Javascript)

1.2 Installation

1.2.1 Install Via Pip(recommended):

```
pip install random-profile   # using pip
conda install random-profile # using anaconda
```

1.2.2 Install from Source(Unreleased):

```
git clone https://github.com/Py-Contributors/RandomProfileGenerator
cd RandomProfileGenerator
python setup.py install
```

1.2.3 Test Installation:

```
random_profile --help
```

1.3 Command line usages

Random Profile Generator can be used as a command line tool. It can be used to generate a random profile and save it to a file. It can also be used to generate a random profile and print it to the console.

Usages:

```
random-profile --help
Usage: random-profile [OPTIONS]

random-profile -n 10 -p

optional arguments:
    -h, --help      show this help message and exit
    -n N            Number of random profiles

    -f, --fullname  Get full name instead of first name
    -p, --profile   Get full profile instead of first name
    -l, --lastname  Get last name instead of first name
    -ip, --ipv4     Get an ipv4 IP address
    -j, --jobtitle  Get job title
```

1.3.1 Get Random Profile:

```
# n = number of random profiles, p = profile
random_profile -n 10 -p
```

1.3.2 Get First Name:

```
# n = number of random profiles, f = first name
random_profile -n 10 -f
```

1.3.3 Get Last Name:

```
# n = number of random profiles, l = last name
random_profile -n 10 -l
```

1.3.4 Get Job Title:

```
# n = number of random profiles, j = job title
random_profile -n 10 -j
```

1.3.5 Get IPv4 Address:

```
# n = number of random profiles, ip = ipv4
random_profile -n 10 -ip
```

1.3.6 Get Random Profile and Save to File:

```
# n = number of random profiles, p = profile
random_profile -n 10 -p > random_profiles.txt
```

1.3.7 Get Random Profile version:

```
random_profile --version

random-profile 0.2.3
```

1.3.8 Get Only Gender Specific Profiles:

To get gender specific profiles, use the *-ma* or *-fe* flags.

```
# n = number of random profiles, p = profile -ma male
random_profile -n 10 -p -ma

# n = number of random profiles, p = profile -fe female
random_profile -n 10 -p -fe
```

1.4 Import as module

You can import the module and use it in your own scripts.

```
from random_profile import RandomProfile
rp = RandomProfile()

    # For first name
rp.first_name(num=10)

# For full name
rp.full_name(num=8)

# override the num value
rp.full_profile(num=10)

# For last name
rp.last_name(num=6)
```

1.5 Run as server

To run as a server, you need to specify the port to listen on:

```
default port is 8000
$ rp --server --port 8080
```

This will start a server on port 8080. You can then use the client to connect to it:

Test it with postman

```
$ curl -X GET http://localhost:8080/ -H 'Content-Type: application/json'
```

Interactive Api Documentation

http://localhost:<port>/docs

1.5.1 API Endpoints

localhost:8000/api/v1/random_profile/full_profile?num=10 localhost:8000/api/v1/random_profile/first_name?num=10 localhost:8000/api/v1/random_profile/last_name?num=10 localhost:8000/api/v1/random_profile/full_name?num=10

1.6 Roadmap and future plans

- *more test coverage*
- *more supported file formats*
- *save book as txt file*

1.7 Run test cases

For the test cases, we are using *pytest*. The test cases are located in the *tests* directory. To run the test cases, you can use the following command:

```
$ pytest tests
```

You can also run the test cases with coverage: