
Sample Data Documentation

Release 0.2.0

Jesús Espino García

March 26, 2015

1	Install	1
2	Quick start	3
3	SampleData	5
3.1	Number methods	5
3.2	Text methods	5
3.3	Time methods	6
3.4	Localized methods	6
3.5	Image methods	7
3.6	Other methods	7
4	Indices and tables	9

Install

Install using pip, including any pillow if you want image generation...:

```
pip install sampledata  
pip install pillow # For image generation
```

Quick start

You can build any kind of object using the `SampleData` class.

```
def generate_random_users(instances):
    sd = SampleData(seed=123)

    users = []
    for x in range(instances):
        data = {
            "slug": sd.slug(2, 3),
            "name": sd.name(2, 3),
            "claim": sd.sentence(),
            "description": sd.paragraph(),
            "email": sd.email(),
            "photo": sd.image_stream(64, 64),
            "is_active": sd.boolean(),
            "birth_date": sd.past_date(),
            "expected_death_date": sd.future_date(),
        }
        users.append(instance)
    return users
```

SampleData

class SampleData (*seed=None*)

SampleData easy the random data generation for a lot of common used data types.

3.1 Number methods

`SampleData.int` (*min_value=0, max_value=sys.maxsize*)

Return an integer between min_value and max_value

`SampleData.number` (*ndigits*)

Return a number of n digits as max

`SampleData.digits` (*ndigits*)

Return a number of exactly n digits

`SampleData.float` (*min, max*)

Return a float from min to max

`SampleData.number_string` (*ndigits*)

Return a string of n digits

3.2 Text methods

`SampleData.char` ()

Return a character between A-Z and a-z

`SampleData.chars` (*min_chars=1, max_chars=5*)

Return a string with n characters between A-Z and a-z being min_chars <= n <= max_chars

`SampleData.word` ()

Returns a lorem ipsum word

`SampleData.words` (*min_words=1, max_words=5*)

Return a string with n lorem ipsum words being min_words <= n <= max_words

`SampleData.email` ()

Return an email

`SampleData.url` ()

Return an url

`SampleData.sentence()`
Return a lorem ipsum sentence (limited to 255 characters)

`SampleData.short_sentence()`
Return a lorem ipsum sentence (limited to 100 characters)

`SampleData.long_sentence()`
Return a lorem ipsum sentence (with 150 characters or more)

`SampleData.paragraph()`
Return a lorem ipsum paragraph

`SampleData.paragraphs(min_paragraphs=1, max_paragraphs=5)`
Return a lorem ipsum text with n paragraphs being $\text{min_paragraphs} \leq n \leq \text{max_paragraphs}$

`SampleData.slug(min_words=5, max_words=5)`
Return a lorem ipsum slug between with n words being $\text{min_words} \leq n \leq \text{max_words}$

`SampleData.tags(min_tags=1, max_tags=5, tags_list=None)`
Return a string of n tags_list or lorem ipsum tags separated by commas being $n \leq \text{min_tags} \leq n \leq \text{max_tags}$

3.3 Time methods

`SampleData.date(begin=-365, end=365)`
Return a date between now+begin and now+end in days

`SampleData.date_between(min_date, max_date)`
Return a date between the min_date and max_date date objects

`SampleData.future_date(min_distance=0, max_distance=365)`
Return a future date between now+min_distance and now+max_distance in days

`SampleData.past_date(min_distance=0, max_distance=365)`
Return a past date between now-max_distance and now-min_distance in days

`SampleData.datetime(begin=-1440, end=1440)`
Return a datetime between now+begin and now+end in minutes

`SampleData.datetime_between(min_datetime, max_datetime)`
Return a datetime between the min_datetime and max_datetime datetime objects

`SampleData.future_datetime(min_distance=0, max_distance=1440)`
Return a future datetime between now+min_distance and now+max_distance in minutes

`SampleData.past_datetime(min_distance=0, max_distance=1440)`
Return a past datetime between now-max_distance and now-min_distance in minutes

`SampleData.time()`
Return a time

3.4 Localized methods

`SampleData.name(locale=None, number=1, as_list=False)`
Return a string or list of typical names from locale using n names (compound names)

Supported locales: cat, es, fr, us

`SampleData.surname(locale=None, number=1, as_list=False)`
Return a string or list of typical surnames from locale using n surnames

Supported locales: cat, es, fr, us

`SampleData.fullname(locale=None, as_list=False)`
Return a string or list of typical names+surnames from locale

Supported locales: cat, es, fr, us

`SampleData.phone(locale, country_code)`
Return a phone number from a country with or without country code

Supported locales: es

`SampleData.zip_code(locale)`
Return a zip code for a country

Supported locales: es

`SampleData.state_code(locale)`
Return a state code for the locale country.

Supported locales: es, us

`SampleData.id_card(locale)`
Return a identification card code for a country

Supported locales: es

3.5 Image methods

`SampleData.image_stream(width, height, typ="simple")`
Return an image of width x height size generated with the typ generator.

Available typ generators: simple, plasma, mandelbrot, ifs, random

`SampleData.image_path_from_directory(directory_path, valid_extensions=['.jpg', '.bmp', '.png'])`
Return an image from a directory with a valid extension

3.6 Other methods

`SampleData.boolean()`
Return a boolean value

`SampleData.nullboolean()`
Return a boolean value or a None

`SampleData.ipv4()`
Return a ipv4 address

`SampleData.ipv6()`
Return a ipv6 address

`SampleData.mac_address()`
Return a mac address

`SampleData.hex_chars(min_chars=1, max_chars=5)`
Return a string with n characters between a-f and 0-9 being min_chars <= n <= max_chars

`SampleData.path` (*absolute=None, extension='', min_levels=1, max_levels=5*)

Return a absolute or relative path (based on *absolute* parameter) string finished in *extension*, and with n levels being $\text{min_levels} \leq n \leq \text{max_levels}$

`SampleData.choice` (*choices*)

Return a value from a list

Indices and tables

- *genindex*
- *search*

B

boolean() (SampleData method), 7

C

char() (SampleData method), 5
chars() (SampleData method), 5
choice() (SampleData method), 8

D

date() (SampleData method), 6
date_between() (SampleData method), 6
datetime() (SampleData method), 6
datetime_between() (SampleData method), 6
digits() (SampleData method), 5

E

email() (SampleData method), 5

F

float() (SampleData method), 5
fullname() (SampleData method), 7
future_date() (SampleData method), 6
future_datetime() (SampleData method), 6

H

hex_chars() (SampleData method), 7

I

id_card() (SampleData method), 7
image_path_from_directory() (SampleData method), 7
image_stream() (SampleData method), 7
int() (SampleData method), 5
ipv4() (SampleData method), 7
ipv6() (SampleData method), 7

L

long_sentence() (SampleData method), 6

M

mac_address() (SampleData method), 7

N

name() (SampleData method), 6
nullboolean() (SampleData method), 7
number() (SampleData method), 5
number_string() (SampleData method), 5

P

paragraph() (SampleData method), 6
paragraphs() (SampleData method), 6
past_date() (SampleData method), 6
past_datetime() (SampleData method), 6
path() (SampleData method), 7
phone() (SampleData method), 7

S

SampleData (built-in class), 5
sentence() (SampleData method), 5
short_sentence() (SampleData method), 6
slug() (SampleData method), 6
state_code() (SampleData method), 7
surname() (SampleData method), 6

T

tags() (SampleData method), 6
time() (SampleData method), 6

U

url() (SampleData method), 5

W

word() (SampleData method), 5
words() (SampleData method), 5

Z

zip_code() (SampleData method), 7