inflection Documentation

Release 0.3.1

Janne Vanhala

Apr 06, 2020

Contents

1 Installation **3**

2 Contributing **5**

3 API Documentation **7**

4 Changelog **11**
4.1 0.3.1 (May 3, 2015) 11

5 License **13**

Python Module Index **15**

Index **17**

Inflection is a string transformation library. It singularizes and pluralizes English words, and transforms strings from CamelCase to underscored_string. Inflection is a port of Ruby on Rails' inflector to Python.

CHAPTER 1

Installation

Use pip to install from PyPI:

```
pip install inflection
```

CHAPTER 2

Contributing

To contribute to Inflector create a fork on GitHub. Clone your fork, make some changes, and submit a pull request.

CHAPTER 3

API Documentation

inflection.**camelize**(*string*, *uppercase_first_letter=True*)

Convert strings to CamelCase.

Examples:

```
>>> camelize("device_type")
"DeviceType"
>>> camelize("device_type", False)
"deviceType"
```

camelize() can be though as a inverse of *underscore()*, although there are some cases where that does not hold:

```
>>> camelize(underscore("IOError"))
"IoError"
```

Parameters **uppercase_first_letter** – if set to *True* *camelize()* converts strings to UpperCamelCase. If set to *False* *camelize()* produces lowerCamelCase. Defaults to *True*.

inflection.**dasherize**(*word*)

Replace underscores with dashes in the string.

Example:

```
>>> dasherize("puni_puni")
"puni-puni"
```

inflection.**humanize**(*word*)

Capitalize the first word and turn underscores into spaces and strip a trailing `"_id"`, if any. Like *titleize()*, this is meant for creating pretty output.

Examples:

```
>>> humanize("employee_salary")
"Employee salary"
>>> humanize("author_id")
"Author"
```

inflection.**ordinal**(*number*)

Return the suffix that should be added to a number to denote the position in an ordered sequence such as 1st, 2nd, 3rd, 4th.

Examples:

```
>>> ordinal(1)
"st"
>>> ordinal(2)
"nd"
>>> ordinal(1002)
"nd"
>>> ordinal(1003)
"rd"
>>> ordinal(-11)
"th"
>>> ordinal(-1021)
"st"
```

inflection.**ordinalize**(*number*)

Turn a number into an ordinal string used to denote the position in an ordered sequence such as 1st, 2nd, 3rd, 4th.

Examples:

```
>>> ordinalize(1)
"1st"
>>> ordinalize(2)
"2nd"
>>> ordinalize(1002)
"1002nd"
>>> ordinalize(1003)
"1003rd"
>>> ordinalize(-11)
"-11th"
>>> ordinalize(-1021)
"-1021st"
```

inflection.**parameterize**(*string*, *separator='-'*)

Replace special characters in a string so that it may be used as part of a 'pretty' URL.

Example:

```
>>> parameterize(u"Donald E. Knuth")
'donald-e-knuth'
```

inflection.**pluralize**(*word*)

Return the plural form of a word.

Examples:

```
>>> pluralize("post")
"posts"
```

(continues on next page)

(continued from previous page)

```
>>> pluralize("octopus")
"octopi"
>>> pluralize("sheep")
"sheep"
>>> pluralize("CamelOctopus")
"CamelOctopi"
```

inflection.**singularize**(*word*)

Return the singular form of a word, the reverse of *pluralize()*.

Examples:

```
>>> singularize("posts")
"post"
>>> singularize("octopi")
"octopus"
>>> singularize("sheep")
"sheep"
>>> singularize("word")
"word"
>>> singularize("CamelOctopi")
"CamelOctopus"
```

inflection.**tableize**(*word*)

Create the name of a table like Rails does for models to table names. This method uses the *pluralize()* method on the last word in the string.

Examples:

```
>>> tableize('RawScaledScorer')
"raw_scaled_scorers"
>>> tableize('egg_and_ham')
"egg_and_hams"
>>> tableize('fancyCategory')
"fancy_categories"
```

inflection.**titleize**(*word*)

Capitalize all the words and replace some characters in the string to create a nicer looking title. *titleize()* is meant for creating pretty output.

Examples:

```
>>> titleize("man from the boondocks")
"Man From The Boondocks"
>>> titleize("x-men: the last stand")
"X Men: The Last Stand"
>>> titleize("TheManWithoutAPast")
"The Man Without A Past"
>>> titleize("raiders_of_the_lost_ark")
"Raiders Of The Lost Ark"
```

inflection.**transliterate**(*string*)

Replace non-ASCII characters with an ASCII approximation. If no approximation exists, the non-ASCII character is ignored. The string must be `unicode`.

Examples:

```
>>> transliterate(u'älämölö')
u'alamolo'
>>> transliterate(u'Ærøskøbing')
u'rskbing'
```

inflection.**underscore**(*word*)

Make an underscored, lowercase form from the expression in the string.

Example:

```
>>> underscore("DeviceType")
"device_type"
```

As a rule of thumb you can think of *underscore()* as the inverse of *camelize()*, though there are cases where that does not hold:

```
>>> camelize(underscore("IOError"))
"IoError"
```

CHAPTER 4

Changelog

Here you can see the full list of changes between each Inflection release.

4.1 0.3.1 (May 3, 2015)

- Fixed trove classifiers not showing up on PyPI.
- Fixed "human" pluralized as "humen" and not "humans".
- Fixed "potato" pluralized as "potatos" and not "potatoes".

4.1.1 0.3.0 (March 1, 2015)

- Added *tableize()* function.

4.1.2 0.2.1 (September 3, 2014)

- Added Python 2, Python 3 and Python 3.4 trove classifiers.

4.1.3 0.2.0 (June 15, 2013)

- Added initial support for Python 3.
- Dropped Python 2.5 support.

4.1.4 0.1.2 (March 13, 2012)

- Added Python 2.5 support.

4.1.5 0.1.1 (February 24, 2012)

- Fixed some files not included in the distribution package.

4.1.6 0.1.0 (February 24, 2012)

- Initial public release

CHAPTER 5

License

Copyright (C) 2012-2015 Janne Vanhala

Permission is hereby granted, free of charge, to any person obtaining a copy of this software and associated documentation files (the “Software”), to deal in the Software without restriction, including without limitation the rights to use, copy, modify, merge, publish, distribute, sublicense, and/or sell copies of the Software, and to permit persons to whom the Software is furnished to do so, subject to the following conditions:

The above copyright notice and this permission notice shall be included in all copies or substantial portions of the Software.

THE SOFTWARE IS PROVIDED “AS IS”, WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM, OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE SOFTWARE.

Python Module Index

i

inflection, 7

Index

C

camelize() (*in module inflection*), 7

D

dasherize() (*in module inflection*), 7

H

humanize() (*in module inflection*), 7

I

inflection (*module*), 7

O

ordinal() (*in module inflection*), 8
ordinalize() (*in module inflection*), 8

P

parameterize() (*in module inflection*), 8
pluralize() (*in module inflection*), 8

S

singularize() (*in module inflection*), 9

T

tableize() (*in module inflection*), 9
titleize() (*in module inflection*), 9
transliterate() (*in module inflection*), 9

U

underscore() (*in module inflection*), 10