PySimpleValidate Documentation

Al Sweigart

Sep 01, 2021

Contents:

A pure-Python module for doing common input validation for Python 2 and 3.

CHAPTER 1

The PySimpleValidate API Reference

CHAPTER 2

Indices and tables

- genindex
- modindex
- search

CHAPTER 3

Installation

```
pip install pysimplevalidate
```

CHAPTER 4

About

PySimpleValidate provides several functions to perform common input validation. The validate* functions in this module accept a *value* argument and raise a *ValidationException* if it doesn't pass validation.

If *value* was valid, the function returns. The return value is the form of the value that the validation function has accepted as value. This could include any transformations such as stripping whitespace from the ends.

The following (hopefully self-descriptive) validation functions are implemented in this module:

- validateNum()
- validateInt()
- validateFloat()
- validateChoice()
- validateDate()
- validateTime()
- validateDatetime()
- validateRegex()
- validateRegexStr()
- validateEmail()
- validateURL()
- validateYesNo()
- validateState()
- validateMonth()
- validateDayOfWeek()
- validateDayOfMonth()

These validation functions have the following common parameters:

- *value*: (str) The value being validated.
- *blank*: (bool) If False, a blank string is considered valid. Defaults to False.
- *strip*: (bool, str, None) If omitted or None, whitespace is stripped from value. If a string, the string's characters are stripped from value. If False, nothing is stripped.
- *allowlistRegexes*: (Sequence, None) A sequence of regex str that will explicitly pass validation, even if they aren't numbers. Defaults to None.
- *blocklistRegexes*: (Sequence, None) A sequence of regex str or (regex_str, response_str) tuples that, if matched, will explicitly fail validation. Defaults to None.

Further, the text-based validators have the following common parameters:

- caseSensitive (bool): If True, value must match the exact casing of an acceptable response. If False, any casing can be used. Defaults to False.

CHAPTER 5

Quickstart

PySimpleValidate's validation functions will raise *ValidationException* if the value passed to them fails validation. Otherwise, a cleaned up version of the value is returned.

It's recommended to import PySimpleValidation with the shorter name `pysv`.

```
>>> import pysimplevalidate as pysv
>>> pysv.validateStr('I have a cat', allowlistRegexes=['caterpillar', 'cat(.*?)dog'],␣
↪blocklistRegexes=['cat', 'm(o){2:}se'])
Traceback (most recent call last):
    ...
pysimplevalidate.ValidationException: This response is invalid.
>>> pysv.validateStr('I have a caterpillar', allowlistRegexes=['caterpillar', 'cat(.*?
↪)dog'], blocklistRegexes=['cat', 'm(o){2:}se'])
'I have a caterpillar'
>>> pysv.validateStr('I have a cat and a dog', allowlistRegexes=['caterpillar', 'cat(.
↪*?)dog'], blocklistRegexes=['cat', 'm(o){2:}se'])
'I have a cat and a dog'
>>> pysv.validateStr('I have a mooooose', allowlistRegexes=['caterpillar', 'cat(.*?
↪)dog'], blocklistRegexes=['cat', 'm(o){2:}se'])
'I have a mooooose'
>>> pysv.validateNum('42')
42
>>> pysv.validateNum('twelve')
Traceback (most recent call last):
    ...
pysimplevalidate.ValidationException: 'twelve' is not a number.
>>>
>>> pysv.validateNum('5', lessThan=10)
5
>>> pysv.validateFloat('4')
4.0
>>> pysv.validateFloat('4.12')
4.12
>>> pysv.validateInt('4.12')
Traceback (most recent call last):
```

(continues on next page)

(continued from previous page)

```
    ...
pysimplevalidate.ValidationException: '4.12' is not an integer.
>>> pysv.validateChoice('cat', ['dog', 'cat', 'moose'])
'cat'
>>> pysv.validateChoice('CAT', ['dog', 'cat', 'moose'])
'cat'
>>> pysv.validateTime('12:00:01')
datetime.time(12, 0, 1)
>>> pysv.validateTime('hour 12 minute 00', formats=['hour %H minute %M'])
datetime.time(12, 0)
>>> pysv.validateEmail('al@inventwithpython.com')
'al@inventwithpython.com'
>>> pysv.validateURL('https://inventwithpython.com')
'https://inventwithpython.com'
>>> pysv.validateYesNo('y')
'yes'
>>> pysv.validateYesNo('NO')
'no'
>>> pysv.validateState('California')
'CA'
>>> pysv.validateState('TEXAS')
'TX'
>>> pysv.validateState('NY')
'NY'
>>> pysv.validateDayOfWeek('mon')
'Monday'
>>> pysv.validateDayOfWeek('FRIday')
'Friday'
>>> pysv.validateDayOfMonth(29, 2004, 2)
29
>>> pysv.validateDayOfMonth(31, 2019, 10)
31
>>> pysv.validateDayOfMonth(31, 2019, 9)
Traceback (most recent call last):
    ...
pysimplevalidate.ValidationException: '31' is not a day in the month of September 2019
```