PyInputPlus Documentation

Al Sweigart

Sep 01, 2021

Contents

PyInputPlus is a Python 3 and 2 module to provide `input()`- and `raw_input()`-like functions with additional validation features. PyInputPlus was created and is maintained by Al Sweigart.

CHAPTER 1

Installation

PyInputPlus can be installed from PyPI using *pip*:

```
pip install pyinputplus
```

On macOS and Linux, installing PyInputPlus for Python 3 is done with *pip3*:

```
pip3 install pyinputplus
```

If you run into permissions errors, try installing with the *–user* option:

```
pip install --user pyinputplus
```

The PySimpleValidate and stdiomask modules will also be installed as a part of PyInputPlus's installation.

CHAPTER 2

Quickstart

PyInputPlus will keep asking the user for text until they enter valid input. It's recommended to import PyInputPlus with the shorter name `pyip`.

```
>>> import pyinputplus as pyip
```

All of PyInputPlus's functions begin with the `input`, such as `inputStr()` or `inputDate()`. Collectively, they are referred to in this documentation as the `input*()` functions.

For example, you can ask the user for an integer with `inputInt()`, and the return value will be an integer instead of the string that `input()` would normally return:

```
>>> input()
42
'42'
>>> response = pyip.inputInt() # keep asking until an int is entered
forty two
'forty two' is not an integer.
42
>>> response
42
```

You could specify a prompt, along with any restrictions you'd like to impose:

```
>>> response = pyip.inputInt('Enter your age: ', min=1)
Enter your age: 0
Number must be at minimum 1.
Enter your age: 2
>>> response
2
```

There are several functions for different common types of data:

```
>>> response = pyip.inputEmail()
alinventwithpython.com
```

(continues on next page)

(continued from previous page)

```
'alinventwithpython.com' is not a valid email address.
al@inventwithpython.com
>>> response
'al@inventwithpython.com'
```

You could also present a small menu of options to the user:

```
>>> response = pyip.inputMenu(['cat', 'dog', 'moose'])
Please select one of the following:
* cat
* dog
* moose
cat
>>> response
'cat'
>>> response = pyip.inputMenu(['cat', 'dog', 'moose'], numbered=True)
Please select one of the following:
1. cat
2. dog
3. moose
1
>>> response
'cat'
```

See the list of functions to get an idea of the kinds of information you can get from the user.

CHAPTER 3

Common input*() Parameters

The following parameters are available for all of the `input*()` functions. You can see this documentation by calling `help(pyip.parameters)`:

```
>>> import pyinputplus as pyip
>>> help(pyip.parameters)
Help on function parameters in module pyinputplus:

parameters()
    Common parameters for all ``input*()`` functions in PyInputPlus:

    * ``prompt`` (str): The text to display before each prompt for user input.␣
↪Identical to the prompt argument for Python's ``raw_input()`` and ``input()``␣
↪functions.
    * ``default`` (str, None): A default value to use should the user time out or␣
↪exceed the number of tries to enter valid input.
    * ``blank`` (bool): If ``True``, a blank string will be accepted. Defaults to␣
↪``False``.
    * ``timeout`` (int, float): The number of seconds since the first prompt for␣
↪input after which a ``TimeoutException`` is raised the next time the user enters␣
↪input.
    * ``limit`` (int): The number of tries the user has to enter valid input before␣
↪the default value is returned.
    * ``strip`` (bool, str, None): If ``None``, whitespace is stripped from value. If␣
↪a str, the characters in it are stripped from value. If ``False``, nothing is␣
↪stripped.
    * ``allowlistRegexes`` (Sequence, None): A sequence of regex str that will␣
↪explicitly pass validation.
    * ``blocklistRegexes`` (Sequence, None): A sequence of regex str or ``(regex_str,␣
↪error_msg_str)`` tuples that, if matched, will explicitly fail validation.
    * ``applyFunc`` (Callable, None): An optional function that is passed the user's␣
↪input, and returns the new value to use as the input.
    * ``postValidateApplyFunc`` (Callable, None): An optional function that is passed␣
↪the user's input after it has passed validation, and returns a transformed version␣
↪for the ``input*()`` function to return.
```

CHAPTER 4

API Reference