argskwargs

Release

Jun 09, 2017

Contents

`argskwargs` is a small Python library that provides a flexible container for positional and keyword arguments.

CHAPTER 1

Installation

```
pip install argskwargs
```

CHAPTER 2

Usage

This tiny library provides a small container class to hold arbitrary positional arguments (`args`) and keyword arguments (`kwargs`). `argskwargs` module is essentially the same as a `(args, kwargs)` tuple, but with a nice and small API on top to keep your code simple and clear.

If you want to pass around `args` and `kwargs` without invoking the function that will eventually end up taking those arguments, one typically uses two variables to hold a `args` tuple and a `kwargs` dict.

The `argskwargs` simplifies this clunky and error-prone code pattern. One can consider `argskwargs` the missing companion to `functools.partial()` from the Python standard library. `argskwargs` lets you treat any combination of `args` and `kwargs` as a single object without attaching it to a function (or another callable), as `functools.partial()` would do.

Here is an example:

```
>>> from argskwargs import argskwargs
>>> my_args = argskwargs(1, 2, foo='bar')
>>> my_args
argskwargs(1, 2, foo='bar')
```

The `my_args` variable simply holds the arguments passed to it. Let's make a function that simply prints out anything that is passed to it:

```
>>> import pprint
>>> def print_arguments(*args, **kwargs):
...     print('positional arguments ' + str(args))
...     print('keyword arguments ' + pprint.pformat(kwargs))
```

You can invoke it directly like this by accessing the `.args` and `.kwargs` attributes (which hold a tuple and a list, respectively):

```
>>> print_arguments(*my_args.args, **my_args.kwargs)
positional arguments (1, 2)
keyword arguments {'foo': 'bar'}
```

Alternatively, you can unpack the `argskwargs` object to obtain a tuple and a dict:

```
>>> x, y = my_args
>>> x
(1, 2)
>>> y
{'foo': 'bar'}
```

So far this is arguably not any better than directly calling `print_arguments(*some_args, **some_kwargs)` using two variables. So, let's see what makes `argskwargs` useful.

Here is another way to do the same using the `.apply()` method:

```
>>> my_args.apply(print_arguments)
positional arguments (1, 2)
keyword arguments {'foo': 'bar'}
```

Since this is the typical use case for `argskwargs`, you can also omit the `.apply()` and call the instance directly:

```
>>> my_args(print_arguments)
positional arguments (1, 2)
keyword arguments {'foo': 'bar'}
```

Magic! As you can see the code is inverted: the callable is passed to the arguments, instead of the other way around.

Time for more magic: assume that we want to pass more arguments to `print_arguments` than those stored in the `argskwargs` instance. Just pass them in:

```
>>> my_args(print_arguments, 'another', oh='yes')
positional arguments (1, 2, 'another')
keyword arguments {'foo': 'bar', 'oh': 'yes'}
```

If you just want to extend the arguments without calling a function, use the `.copy()` method, which does exactly that:

```
>>> more_args = my_args.copy(3, 4, abc='xyz')
>>> more_args
argskwargs(1, 2, 3, 4, abc='xyz', foo='bar')
>>> more_args(print_arguments)
positional arguments (1, 2, 3, 4)
keyword arguments {'abc': 'xyz', 'foo': 'bar'}
```

As descibed above, `argskwargs` is a companion to `functools.partial()`. You can directly create partial functions using the `.partial()` method:

```
>>> f = my_args.partial(print_arguments)
>>> f()
positional arguments (1, 2)
keyword arguments {'foo': 'bar'}
```

For completeness, you can pass more arguments in one go, but you may want to avoid that for your own sanity:

```
>>> g = my_args.partial(print_arguments, 'insane', foo='foofoo')
>>> g()
positional arguments (1, 2, 'insane')
keyword arguments {'foo': 'foofoo'}
```

Hopefully this demonstrated the usefulness of `argskwargs`, so by all means use it, but please keep this mantra in mind: *simple is better than complex*.

Happy hacking!

CHAPTER 3

API

CHAPTER 4

Contributing

The source code and issue tracker for this package can be found on Github:

https://github.com/wbolster/argskwargs

CHAPTER 5

License

..include:: ../LICENSE.rst