Class Namespaces Documentation

Release 0.3.7

Max Woerner Chase

Feb 23, 2021

Contents

Class Namespaces is a library for Python 3.3+, that allows you to break your Python classes into distinct namespaces. If your classes have a bunch of responsibilities, cushion the blow a bit by dividing methods, attributes, and descriptors into those responsibilities.

Contents:

CHAPTER 1

Namespaces

Class Namespaces provides a small collection of types. Neither type has public methods of interest; they are used solely in the context of Python's own protocols.

1.1 NamespaceableMeta

class class_namespaces.**NamespaceableMeta**

Metaclass for classes that can contain namespaces.

A note for people extending the functionality: The base class for NamespaceableMeta uses a non-standard super() invocation in its definitions of several methods. This was the only way I could find to mitigate some bugs I encountered with a standard invocation. If you override any of methods defined on built-in types, I recommend this form for maximal reusability:

super(class, type(self)).__method__(self, . . .)

1.2 Namespaceable

class class_namespaces.**Namespaceable**

Optional convenience class. Inherit from it to get the metaclass.

1.3 Namespace

class class_namespaces.**Namespace**(**args*, ***kwargs*)

Namespace.

Namespace() -> new empty namespace Namespace(mapping) -> new namespace initialized from a mapping object's

(key, value) pairs

Namespace(iterable) -> new namespace initialized as if via: d = {} for k, v in iterable:

d[k] = v

ns = Namespace(d)

Namespace(kwargs) -> new namespace initialized with the name=value pairs** in the keyword argument list. For example: Namespace(one=1, two=2)

Namespaces implement the context manager protocol. When the context is entered in an appropriate class creation scope, the Namespace shadows the currently visible scope.

Namespaces can be re-entered after they are exited, provided they're in the same parent scope.

CHAPTER 2

Compatibility

Class Namespaces provides compatibility modules for using other metaclasses at the same time. Currently, the only provided module is compatibility for Abstract Base Classes. Use the provided classes, together with the abc module's tools, such as the abstractmethod decorator.

CHAPTER 3

Indices and tables

- genindex
- modindex
- search

Index

N