pimpmyclass Documentation

Release 0.4.3

Hernan E. Grecco

Apr 30, 2019

Contents:

Pimp your class with awesome features

This library provides base classes to enable useful behavior in Python Objects.

The central purpose of the library is to extend python properties to allow:

- get/set logging.
- get/set timing, success and failure stats.
- async locking.
- get/set coercion and conversion.
- value cache
- prevent unnecessary set.
- read once properties

But most importantly, it allows owner specific configurations. Properties are class attributes, and therefore it is difficult to have a property which is, for exampled cached, in an object but not cached in another instance of the same class.

The library also provides DictProperties: that is properties that can be accessed by key; and also methods!

Each capability is isolated in individual classes allowing you to pick only what you need.

Follow us on GitHub: https://github.com/hgrecco/pimpmyclass

CHAPTER 1

Basic usage

Pick what capability you need and created your pimped property

```
from pimpmyclass import props

class MyAwesomeProperty(props.StatsProperty):
    """Docs goes here
    """
```

Check what is required for that class and define your base class with your properties.

```
from pimpmyclass import mixins

class Base(mixins.StorageMixin):
    """Docs goes here
    """

    @MyAwesomeProperty()
    def just_read(self):
        return 42

    @MyAwesomeProperty()
    def read_write(self):
        return 42

    @read_write.setter
    def read_write(self, value):
        if value != 42:
            raise ValueError
```

and that's it!

```
>>> obj = Base()
>>> obj.just_read
42
```

(continues on next page)

(continued from previous page)

```
>>> obj.just_read
42
>>> obj.read_write
42
>>> obj.read_write = 42
>>> obj.read_write = 43
Traceback (most recent call last):
...
ValueError
>>> s = Base.just_read.stats(obj, 'get')
>>> s.count # number of times get was called
2
>>> s.last # duration in seconds of the last get call (you can also ask for mean, max,
↪ min, std)
1.414009602740407e-06
>>> Base.read_write.stats(obj, 'failed_set').count
1
```

1.1 Properties

class NamedProperty(*fget=None*, *fset=None*, *fdel=None*, *doc=None*, ***kwargs*)

Bases: `pimpmyclass.common.NamedCommon`

A property that takes the name of the class attribute to which it is assigned.

Accepts instance of `pimpmyclass.Config` as configuration values that automatically get populated from kwargs.

class StorageProperty(*fget=None*, *fset=None*, *fdel=None*, *doc=None*, ***kwargs*)

Bases: *pimpmyclass.props.NamedProperty*

A property that can store and retrieve information in the instance to which is attached.

Requires that the owner class inherits *pimpmyclass.mixins.StorageMixin*.

Notes

The information is stored in uniquely a specified namespace defined by the derived class. Inside that storage, another namespace is specified using the property name.

Derived class should use the dynamically created _store_get and _store_set to retrieve and store information.

Note: Derived classes must override the following variables:

_storage_ns [str] Defines a unique namespace under which the information of the derived class is stored.

_storage_ns_init [callable] Called upon initialization of the storage to initialize the specific storage of the namespace.

---

class StatsProperty(*fget=None*, *fset=None*, *fdel=None*, *doc=None*, ***kwargs*)

Bases: *pimpmyclass.props.StorageProperty*

A property that keep stats on get and set calls.

Stats can be retrieved with the *stat* methods and the following keys:

- get
- set
- failed_get
- failed_set

The following statistics are provided in a namedtuple:

last [float] most recent duration (seconds).

count [int] number of operations.

mean [float] average duration per operation (seconds).

std [float] standard deviation of the duration (seconds).

min [float] shortest duration (seconds).

max [float] longest duration (seconds).

Requires that the owner class inherits *pimpmyclass.mixins.StorageMixin*.

class LogProperty (*fget=None*, *fset=None*, *fdel=None*, *doc=None*, ***kwargs*)
Bases: *pimpmyclass.props.NamedProperty*

A property that log operations.

Requires that the owner class inherits *pimpmyclass.mixins.LogMixin*.

log_values [(default=True)] If False, just the types (not the values) will logged.

An obj to str callable can be provided to perform custom serialization.

class LockProperty (*fget=None*, *fset=None*, *fdel=None*, *doc=None*, ***kwargs*)
Bases: *pimpmyclass.props.NamedProperty*

A property that with a set or get Lock.

Requires that the owner class inherits *pimpmyclass.mixins.LogMixin*.

class TransformProperty (*fget=None*, *fset=None*, *fdel=None*, *doc=None*, ***kwargs*)
Bases: pimpmyclass.props.InstanceConfigurableProperty

A property that can transform value before a set operation or after a get operation.

Requires that the owner class inherits pimpmyclass.mixins.InstanceConfigurableProperty.

Other Parameters

- **pre_set** (*Note: checking function (default=None)*)
- **post_get** (*Note: checking function (default=None)*)

class CacheProperty (*fget=None*, *fset=None*, *fdel=None*, *doc=None*, ***kwargs*)
Bases: *pimpmyclass.props.StorageProperty*

A property that can store, recall or invalidate a cache.

class GetCacheProperty (*fget=None*, *fset=None*, *fdel=None*, *doc=None*, ***kwargs*)
Bases: *pimpmyclass.props.CacheProperty*

A property that stores the get value in the cache.

Requires that the owner class inherits *pimpmyclass.mixins.StorageMixin*.

class SetCacheProperty(*fget=None*, *fset=None*, *fdel=None*, *doc=None*, ***kwargs*)
Bases: *pimpmyclass.props.CacheProperty*

A property that stores the set value in the cache.

Requires that the owner class inherits *pimpmyclass.mixins.StorageMixin*.

class PreventUnnecessarySetProperty(*fget=None*, *fset=None*, *fdel=None*, *doc=None*, ***kwargs*)
Bases: *pimpmyclass.props.SetCacheProperty*

A property that prevents unnecessary set operations by comparing the value in the cache with the value to be set.

Requires that the owner class inherits *pimpmyclass.mixins.CacheMixin* and *pimpmyclass.mixins.LogMixin*.

class ReadOnceProperty(*fget=None*, *fset=None*, *fdel=None*, *doc=None*, ***kwargs*)
Bases: pimpmyclass.props.InstanceConfigurableProperty, *pimpmyclass.props.GetCacheProperty*

Avoids calling the getter if the value is already in the cache.

Other Parameters **read_once** (*bool (default=False)*)

class ObservableProperty(*fget=None*, *fset=None*, *fdel=None*, *doc=None*, ***kwargs*)
Bases: *pimpmyclass.props.CacheProperty*

A property that emits a signal when the cached value is changed (either via set or get)

Requires that the owner class inherits *pimpmyclass.mixins.CacheMixin* and *pimpmyclass.mixins.ObservableMixin*.

1.2 Methods

class NamedMethod(***kwargs*)
Bases: pimpmyclass.common.NamedCommon

class StorageMethod(***kwargs*)
Bases: *pimpmyclass.methods.NamedMethod*

A property that can store and retrieve information in the instance to which is attached.

Methods and descriptors are class attributes and therefore any attempt to naively modify one of their attributes for a single instance of the parent class will propagate to all instances. This property overcomes this problem by storing information at the instance level.

The information is stored in uniquely a specified namespace defined by the derived class. Inside that storage, another namespace is specified using the property name.

Derived class should use the dynamically created _store_get and _store_set to retrieve and store information.

..note: Derived classes must override the following variables:

_storage_ns [str] Defines a unique namespace under which the information of the derived class is stored.

_storage_ns_init [callable] Called upon initialization of the storage to initialize the specific storage of the namespace.

Requires that the owner class inherits *pimpmyclass.mixins.StorageMixin*.

class StatsMethod(***kwargs*)
Bases: *pimpmyclass.methods.StorageMethod*

A property that keep stats on get and set calls.

Stats can be retrieved with the *stat* methods and the following keys: - call - failed_call

The following statistics are provided in a namedtuple:

last [float] most recent duration (seconds).

count [int] number of operations.

mean [float] average duration per operation (seconds).

std [float] standard deviation of the duration (seconds).

min [float] shortest duration (seconds).

max [float] longest duration (seconds).

Requires that the owner class inherits *pimpmyclass.mixins.StorageMixin*.

class LogMethod(***kwargs*)
Bases: *pimpmyclass.methods.NamedMethod*

Other Parameters **log_values** (*(default=True)*)

class LockMethod(***kwargs*)
Bases: *pimpmyclass.methods.NamedMethod*

class InstanceConfigurableMethod(***kwargs*)
Bases: *pimpmyclass.methods.StorageMethod*

class TransformMethod(***kwargs*)
Bases: *pimpmyclass.methods.InstanceConfigurableMethod*

Other Parameters **params** (*(default=None)*)

classmethod param(*names*, *func*)
Add modifiers to a specific parameter.

See Action for more information.

classmethod ret(*func*)
Add modifiers to the return value.

See Action for more information.

1.3 Mixins

class StorageMixin
Bases: `object`

Mixin class to be inherited by a class that uses a StorageProperty.

Provides an instance specific storage space divided into namespaces.

storage
Storage for the instance.

Returns

Return type dict-like object

class BaseLogMixin

Bases: object

Base Mixin class to be inherited by a class requiring logging.

Derived classes can specify the logger_name by overriding the logger name variable.

Optionally, an instance of logging.Logger can be attached after instantiation using the logger attribute.

Extra information to each log record can be added permanently using the logger extra attribute.

log(*level*, *msg*, **args*, ***kwargs*)

Log with the integer severity 'level' on the logger corresponding to this class.

Must be implemented by classes actually logging something.

Parameters

- **level** – severity level for this event.
- **msg** – message to be logged (can contain PEP3101 formatting codes)
- ***args** – arguments passed to logger.log
- ****kwargs** – keyword arguments passed to the logger.log

See also:

log_info(), *log_debug()*, *log_error()*, *log_warning()*, *log_critical()*

log_critical(*msg*, **args*, ***kwargs*)

Log with the severity 'CRITICAL' on the logger corresponding to this instance.

See also:

log(), *log_info()*, *log_debug()*, *log_error()*, *log_warning()*

log_debug(*msg*, **args*, ***kwargs*)

Log with the severity 'DEBUG' on the logger corresponding to this instance.

See also:

log(), *log_info()*, *log_error()*, *log_warning()*, *log_critical()*

log_error(*msg*, **args*, ***kwargs*)

Log with the severity 'ERROR' on the logger corresponding to this instance.

See also:

log(), *log_info()*, *log_debug()*, *log_warning()*, *log_critical()*

log_info(*msg*, **args*, ***kwargs*)

Log with the severity 'INFO' on the logger corresponding to this instance.

See also:

log(), *log_debug()*, *log_error()*, *log_warning()*, *log_critical()*

log_warning(*msg*, **args*, ***kwargs*)

Log with the severity 'WARNING' on the logger corresponding to this instance.

See also:

log(), *log_info()*, *log_debug()*, *log_error()*, *log_critical()*

class LogMixin

Bases: *pimpmyclass.mixins.BaseLogMixin*

Mixin class to be inherited by a class requiring logging to Python std logging.

Derived classes can specify the logger_name by overriding the logger name variable.

Optionally, an instance of logging.Logger can be attached after instantiation using the logger attribute.

Extra information to each log record can be added permanently using the logger extra attribute.

log(*level*, *msg*, **args*, ***kwargs*)
Log with the integer severity 'level' on the logger corresponding to this class.

Parameters

- **level** – severity level for this event.
- **msg** – message to be logged (can contain PEP3101 formatting codes)
- ***args** – arguments passed to logger.log
- ****kwargs** – keyword arguments passed to the logger.log

See also:

`log_info()`, `log_debug()`, `log_error()`, `log_warning()`, `log_critical()`

class LockMixin
Bases: `object`

Mixin class to be inherited by a class requiring a reentrant lock.

class AsyncMixin
Bases: *pimpmyclass.mixins.LockMixin*

Mixin class to be inherited by a class requiring async operations.

Async operations are implemented using a single worker Thread Pool Executor that returns futures.

The number of pending tasks can be found in *async_pending*.

class CacheMixin
Bases: `object`

recall(*keys*)
Return the last value seen for a CacheProperty or a collection of CacheProperties.

Parameters **keys** (*str or iterable of str, optional*) – Name of the CacheProperty or properties to recall.

Returns

Return type value of the CacheProperty or dict mapping CacheProperty to values.

class ObservableMixin
Bases: `object`

1.4 DictProperties

class DictProperty(**args*, ***kwargs*)
Bases: *pimpmyclass.props.NamedProperty*

class DictCacheProperty(**args*, ***kwargs*)
Bases: *pimpmyclass.dictprops.DictProperty*

class DictObservableProperty(**args*, ***kwargs*)
Bases: *pimpmyclass.dictprops.DictCacheProperty*

CHAPTER 2

Indices and tables

- genindex
- modindex
- search

Index