
link.feature Documentation

Release 2.1

David Delassus

September 02, 2016

1	Installation	3
2	Contents	5
2.1	Tutorial	5
2.2	API documentation	6
3	Donating	9
	Python Module Index	11

link.feature is used to make classes providing features.

Check out the source code on [Github](#).

Installation

```
pip install link.feature
```

Contents

2.1 Tutorial

This package is used to make classes providing features. A feature is a class, providing an API using the class' instance.

For example:

```
from link.feature import Feature
import json

class JSONFeature(Feature):
    name = 'json'

    def to_json(self):
        return json.dumps(self.obj.data)
```

Then, we just have to add the feature to the class, using the decorator:

```
from link.feature import addfeatures

@addfeatures([JSONFeature])
class MyClass(object):
    def __init__(self, *args, **kwargs):
        super(MyClass, self).__init__(*args, **kwargs)

        self.data = {'foo': 'bar'}
```

You can now get a list of all features provided by an instance:

```
from link.feature import getfeatures

obj = MyClass()
result = getfeatures(obj)

# result == [(obj, JSONFeature)]
```

NB: `getfeatures()` look for featured properties (a special kind of property, used to indicate that there may be resolvable features):

```
from link.feature import featuredprop
```

```
class Dummy(object):
    def __init__(self, *args, **kwargs):
        super(Dummy, self).__init__(*args, **kwargs)

        self._inner = MyClass()

    @featuredprop
    def inner(self):
        return self._inner

obj2 = Dummy()
result = getfeatures(obj2)

# result == [(obj2.inner, JSONFeature)]
```

Now, you can check if an object provides a feature, and instantiate it:

```
from link.feature import hasfeature, getfeature

assert hasfeature(obj, 'json')

try:
    f = getfeature(obj, 'json')
except AttributeError:
    print('obj has no feature json')

result = f.to_json()
```

2.2 API documentation

2.2.1 link.feature package

Module contents

class `link.feature.Feature` (*obj*, **args*, ***kwargs*)
Bases: `object`

Base class for feature.

Parameters `obj` (*any*) – Object providing this feature

name = `None`

class `link.feature.featuredprop`
Bases: `property`

Special property to allow traversal by the `getfeatures()` function.

`link.feature.addfeatures` (*features*)
Add features to a class.

Parameters `features` (*list*) – Features to add

Returns decorator

Return type callable

`link.feature.get_features(obj)`

Get list of features provided by an object's class.

Parameters `obj` (*any*) – Object

Returns List of provided features

Return type list

`link.feature.has_feature(obj, name)`

Check if object's class provides a feature.

Parameters

- `obj` (*any*) – Object
- `name` (*str*) – Feature's name

Returns True if feature is provided

Return type bool

`link.feature.get_feature(obj, name, *args, **kwargs)`

Instantiate a feature.

Parameters

- `obj` (*any*) – Object
- `name` (*str*) – Feature's name
- `args` (*Iterable*) – Positional arguments for feature's constructor
- `kwargs` (*dict*) – Keyword arguments for feature's constructor

Returns Instance of feature

Return type *Feature*

Raises `AttributeError` – if feature is not provided by object's class

Donating

|

`link.feature`, 6

A

`addfeatures()` (in module `link.feature`), 6

F

`Feature` (class in `link.feature`), 6

`featuredprop` (class in `link.feature`), 6

G

`getfeature()` (in module `link.feature`), 7

`getfeatures()` (in module `link.feature`), 6

H

`hasfeature()` (in module `link.feature`), 7

L

`link.feature` (module), 6

N

`name` (`link.feature.Feature` attribute), 6