python-loaders Documentation

Release 0.2.3

Oskari Hiltunen

October 08, 2015

Contents

1 Installation 3

2 API Reference 5

Python Module Index 7

Python-Loaders is a small collection of custom module loaders doing various tasks.

CHAPTER 1

Installation

```
pip install python-loaders
```

CHAPTER 2

API Reference

class `loaders.`**`Lazy`**(*module_name*, *attrs*)

Prevents the module from being loaded until the lazy loader's *`ready()`* method has been called. The module can still be imported and the attrs defined in `attrs` can still be accessed before *`ready()`* has been called, and references to them will be updated with the real values when the method is called.

This can be useful when your module has some requirements that cannot be satisfied when the module would be first imported. One example of this is defining a class whose parent is given as a parameter to the package's setup function:

`pkg/__init__.py`:

```
import loaders
loader = loaders.Lazy('%s.models' % __name__, ['MyModel'])
import extensions


def init(db):
    extensions.db = db
    loader.ready()
```

`pkg/extensions.py`:

```
db = None
```

`pkg/models.py`:

```
from .extensions import db


# db is None until the init function has been called, so loading this
# module before that would result in an AttributeError.
class MyModel(db.Model):
    name = db.Column(db.Unicode(255))
```

Parameters

- **`module_name`** (*string*) – The full name of the module to make lazy. For example `"pkg.models"`.
- **`attrs`** (*iterable of strings*) – An iterable object containing attributes that should be exportable from the module before the module has been loaded. The others will still be importable after *`ready()`* has been called.

ready()

Loads the module and updates references to it and the attributes defined in `attrs`. Call this when the module's requirements have been set up and the module can be loaded.

Python Module Index

l

loaders, 5

Index

L

Lazy (class in loaders), 5
loaders (module), 5

R

ready() (loaders.Lazy method), 5