
zict Documentation

Release 0.0.2

Matthew Rocklin

September 03, 2016

1 Example	3
2 API	5

The dictionary / mutable mapping interface is powerful and multi-faceted.

- We store data in different locations such as in-memory, on disk, in archive files, etc..
- We manage old data with different policies like LRU, random eviction, etc..
- We might encode or transform data as it arrives or departs the dictionary through compression, encoding, etc..

To this end we build abstract `MutableMapping` classes that consume and build on other `MutableMappings`. We can compose several of these with each other to form intuitive interfaces over complex storage systems policies.

Example

In the following example we create an LRU dictionary backed by pickle-encoded, zlib-compressed, directory of files.

```
import pickle
import zlib

from zict import File, Func, LRU

a = File('myfile/', mode='a')
b = Func(zlib.compress, zlib.decompress, a)
c = Func(pickle.dumps, pickle.loads, b)
d = LRU(100, c)

>>> d['x'] = [1, 2, 3]
>>> d['x']
[1, 2, 3]
```

class `zict.lru.LRU` (*n*, *d*, *on_evict*=<function *do_nothing*>, *weight*=<function <lambda>>)
Evict Least Recently Used Elements

Parameters *n*: int

Number of elements to keep, or total weight if *weight*= is used

d: MutableMapping

Dictionary in which to hold elements

on_evict: callable

Function:: *k*, *v* -> action to call on key value pairs prior to eviction

weight: callable

Function:: *k*, *v* -> number to determine the size of keeping the item in the mapping.
Defaults to (*k*, *v*) -> 1

Examples

```
>>> lru = LRU(2, dict(), on_evict=lambda k, v: print("Lost", k, v))
>>> lru['x'] = 1
>>> lru['y'] = 2
>>> lru['z'] = 3
Lost x 1
```

Methods

class `zict.file.File` (*directory*, *mode*='a')
Mutable Mapping interface to a directory

Keys must be strings, values must be bytes

Parameters *directory*: string

mode: string, ('r', 'w', 'a'), defaults to 'a'

Examples

```
>>> z = File('myfile')
>>> z['x'] = b'123'
>>> z['x']
b'123'
```

Methods

class `zict.func.Func` (*dump, load, d*)
Buffer a MutableMapping with a pair of input/output functions

Parameters **dump:** callable

Function to call on value as we set it into the mapping

load: callable

Function to call on value as we pull it from the mapping

d: MutableMapping

Examples

```
>>> def double(x):
...     return x * 2
```

```
>>> def halve(x):
...     return x / 2
```

```
>>> d = dict()
>>> f = Func(double, halve, d)
>>> f['x'] = 10
>>> d
{'x': 20}
>>> f['x']
10.0
```

Methods

class `zict.zip.Zip` (*filename, mode='a'*)
Mutable Mapping interface to a Zip file

Keys must be strings, values must be bytes

Parameters **filename:** string

mode: string, ('r', 'w', 'a'), defaults to 'a'

Examples

```
>>> z = Zip('myfile.zip')
>>> z['x'] = b'123'
>>> z['x']
```

```
b'123'  
>>> z.flush() # flush and write metadata to disk
```

Attributes

Methods

F

File (class in zict.file), 5
Func (class in zict.func), 6

L

LRU (class in zict.lru), 5

Z

Zip (class in zict.zip), 6