defaultdict — Missing Keys Return a Default Value

The standard dictionary includes the method `setdefault()` for retrieving a value and establishing a default if the value does not exist. By contrast, `defaultdict` lets the caller specify the default up front when the container is initialized.

```
# collections_defaultdict.py

import collections


def default_factory():
    return 'default value'


d = collections.defaultdict(default_factory, foo='bar')
print('d:', d)
print('foo =>', d['foo'])
print('bar =>', d['bar'])
```

This method works well as long as it is appropriate for all keys to have the same default. It can be especially useful if the default is a type used for aggregating or accumulating values, such as a `list`, `set`, or even `int`. The standard library documentation includes several examples in which `defaultdict` is used in this way.

```
$ python3 collections_defaultdict.py

d: defaultdict(<function default_factory at 0x101341950>,
{'foo': 'bar'})
foo => bar
bar => default value
```

See also

- defaultdict examples – Examples of using `defaultdict` from the standard library documentation.
- Evolution of Default Dictionaries in Python – James Tauber’s discussion of how `defaultdict` relates to other means of initializing dictionaries.

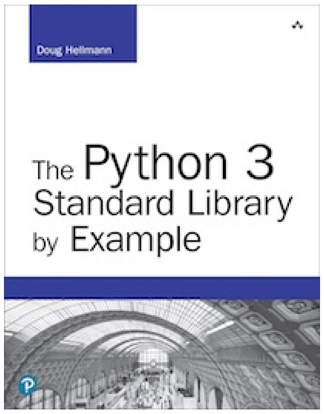
Counter — Count Hashable Objects

deque — Double-Ended Queue

This page was last updated 2017-01-28.

Navigation

Counter — Count Hashable Objects
deque — Double-Ended Queue



The output from all the example programs from PyMOTW-3 has been generated with Python 3.7.1, unless otherwise noted. Some of the features described here may not be available in earlier versions of Python.

Looking for examples for Python 2?

This Site

Module Index
Index

Other Writing

Blog
The Python Standard Library By Example