

---

# **FlatDict Documentation**

***Release 1.1.1***

**Gavin M. Roy**

**Mar 07, 2018**



---

## Contents

---

<b>1</b>	<b>Installation</b>	<b>3</b>
<b>2</b>	<b>Example Use</b>	<b>5</b>
<b>3</b>	<b>Class Documentation</b>	<b>7</b>
	<b>Python Module Index</b>	<b>11</b>



**FlatDict** is a Python module for interacting with nested dicts as a single level dict with delimited keys. *FlatDict* supports Python 2.6+ and 3.2+.

Jump to [Installation](#), [Example Use](#), [Class Documentation](#), or [license](#).

*For example:*

```
foo = {'foo': {'bar': 'baz', 'qux': 'corge'}}
```

*is represented as:*

```
{'foo:bar': 'baz',  
 'foo:qux': 'corge'}
```

*And can still be accessed as:*

```
foo['foo']['bar']
```

*and*

```
foo['foo:bar']
```

Additionally, lists and tuples are also converted into dicts using `enumerate()`.

*For example:*

```
d = {'list': ['a', 'b', 'c', ]}
```

*Will be flattened as follows:*

```
flat = {'list:0': 'a', 'list:1': 'b', 'list:2': 'c'}
```



# CHAPTER 1

---

## Installation

---

```
$ pip install flatdict
```





## CHAPTER 2

---

### Example Use

---

```
import flatdict

values = {'foo': {'bar': {'baz': 0,
                        'qux': 1,
                        'corge': 2},
          'grault': {'baz': 3,
                    'qux': 4,
                    'corge': 5}},
         'garply': {'foo': 0, 'bar': 1, 'baz': 2, 'qux': {'corge': 3}}}}

flat = flatdict.FlatDict(values)

print flat['foo:bar:baz']

flat['test:value:key'] = 10

del flat['test']

for key in flat:
    print key

for values in flat.itervalues():
    print key

print repr(flat.as_dict())
```



---

Class Documentation

---

FlatDict is a dict object that allows for single level, delimited key/value pair mapping of nested dictionaries.

**class** flatdict.**FlatDict** (*value=None, delimiter=None, former\_type=<type 'dict'>*)

*FlatDict* is a dictionary object that allows for single level, delimited key/value pair mapping of nested dictionaries. The default delimiter value is `:` but can be changed in the constructor or by calling *FlatDict.set\_delimiter*.

**DELIMITER** = `':'`

**as\_dict** ()

Return the flat dictionary as a dictionary.

**Return type** dict

**clear** ()

Remove all items from the flat dictionary.

**copy** ()

Return a shallow copy of the flat dictionary.

**Return type** *flatdict.FlatDict*

**fromkeys** (*S*, [*v*]) → New dict with keys from *S* and values equal to *v*.  
*v* defaults to None.

**get** (*key*, *d=None*)

Return the value for *key* if *key* is in the flat dictionary, else default. If default is not given, it defaults to None, so that this method never raises a `KeyError`.

**Parameters**

- **key** (*mixed*) – The key to get
- **d** (*mixed*) – The default value

**Return type** mixed

**has\_key** (*key*)

Check to see if the flat dictionary has a specific key.

**Parameters** **key** (*mixed*) – The key to check for

**Return type** bool

**items** ()

Return a copy of the flat dictionary's list of (key, value) pairs.

---

**Note:** CPython implementation detail: Keys and values are listed in an arbitrary order which is non-random, varies across Python implementations, and depends on the flat dictionary's history of insertions and deletions.

---

**Return type** list

**iteritems** ()

Return an iterator over the flat dictionary's (key, value) pairs. See the note for `FlatDict.items()`.

Using `iteritems()` while adding or deleting entries in the flat dictionary may raise a `RuntimeError` or fail to iterate over all entries.

**Return type** Iterator

**Raises** `RuntimeError`

**iterkeys** ()

Return an iterator over the flat dictionary's keys. See the note for `FlatDict.items()`.

Using `iterkeys()` while adding or deleting entries in the flat dictionary may raise a `RuntimeError` or fail to iterate over all entries.

**Return type** Iterator

**Raises** `RuntimeError`

**intervalues** ()

Return an iterator over the flat dictionary's values. See the note for `FlatDict.items()`.

Using `intervalues()` while adding or deleting entries in the flat dictionary may raise a `RuntimeError` or fail to iterate over all entries.

**Return type** Iterator

**Raises** `RuntimeError`

**keys** ()

Return a copy of the flat dictionary's list of keys. See the note for `FlatDict.items()`.

**Return type** list

**pop** (*key*, *default=None*)

If key is in the flat dictionary, remove it and return its value, else return default. If default is not given and key is not in the dictionary, a `KeyError` is raised.

**Parameters**

- **key** (*mixed*) – The key name
- **default** (*mixed*) – The default value

**Return type** mixed

**popitem** () → (k, v), remove and return some (key, value) pair as a 2-tuple; but raise `KeyError` if D is empty.

**set\_delimiter** (*delimiter*)

Override the default or passed in delimiter with a new value.

**Parameters** **delimiter** (*str*) – The delimiter to use

**setdefault** (*key*, *default*)

If key is in the flat dictionary, return its value. If not, insert key with a value of default and return default. default defaults to `None`.

**Parameters**

- **key** (*mixed*) – The key name
- **default** (*mixed*) – The default value

**Return type** `mixed`

**update** (*other=None*, *\*\*kwargs*)

Update the flat dictionary with the key/value pairs from other, overwriting existing keys.

`update()` accepts either another flat dictionary object or an iterable of key/value pairs (as tuples or other iterables of length two). If keyword arguments are specified, the flat dictionary is then updated with those key/value pairs: `d.update(red=1, blue=2)`.

**Return type** `None`

**values** ()

Return a copy of the flat dictionary's list of values. See the note for `FlatDict.items()`.

**Return type** `list`

**viewitems** () → a set-like object providing a view on D's items

**viewkeys** () → a set-like object providing a view on D's keys

**viewvalues** () → an object providing a view on D's values



**f**

`flatdict`, [7](#)





## A

`as_dict()` (`flatdict.FlatDict` method), 7

## C

`clear()` (`flatdict.FlatDict` method), 7

`copy()` (`flatdict.FlatDict` method), 7

## D

`DELIMITER` (`flatdict.FlatDict` attribute), 7

## F

`FlatDict` (class in `flatdict`), 7

`flatdict` (module), 7

`fromkeys()` (`flatdict.FlatDict` method), 7

## G

`get()` (`flatdict.FlatDict` method), 7

## H

`has_key()` (`flatdict.FlatDict` method), 7

## I

`items()` (`flatdict.FlatDict` method), 8

`iteritems()` (`flatdict.FlatDict` method), 8

`iterkeys()` (`flatdict.FlatDict` method), 8

`itervalues()` (`flatdict.FlatDict` method), 8

## K

`keys()` (`flatdict.FlatDict` method), 8

## P

`pop()` (`flatdict.FlatDict` method), 8

`popitem()` (`flatdict.FlatDict` method), 8

## S

`set_delimiter()` (`flatdict.FlatDict` method), 8

`setdefault()` (`flatdict.FlatDict` method), 9

## U

`update()` (`flatdict.FlatDict` method), 9

## V

`values()` (`flatdict.FlatDict` method), 9

`viewitems()` (`flatdict.FlatDict` method), 9

`viewkeys()` (`flatdict.FlatDict` method), 9

`viewvalues()` (`flatdict.FlatDict` method), 9