12.10. dbhash — DBM-style interface to the BSD database library¶

Deprecated since version 2.6: The `dbhash` module has been deprecated for removal in Python 3.0.

The `dbhash` module provides a function to open databases using the BSD db library. This module mirrors the interface of the other Python database modules that provide access to DBM-style databases. The `bsddb` module is required to use `dbhash`.

This module provides an exception and a function:

exception `dbhash.error`¶

Exception raised on database errors other than `KeyError`. It is a synonym for `bsddb.error`.

`dbhash.open`(*path*[, *flag*[, *mode*]])¶

Open a db database and return the database object. The *path* argument is the name of the database file.

The *flag* argument can be:

Value	Meaning
`'r'`	Open existing database for reading only (default)
`'w'`	Open existing database for reading and writing
`'c'`	Open database for reading and writing, creating it if it doesn't exist
`'n'`	Always create a new, empty database, open for reading and writing

For platforms on which the BSD db library supports locking, an `'l'` can be appended to indicate that locking should be used.

The optional *mode* parameter is used to indicate the Unix permission bits that should be set if a new database must be created; this will be masked by the current umask value for the process.

See also

Module `anydbm`

Generic interface to `dbm`-style databases.

Module `bsddb`

Lower-level interface to the BSD db library.

Module `whichdb`

Utility module used to determine the type of an existing database.

12.10.1. Database Objects¶

The database objects returned by `open()` provide the methods common to all the DBM-style databases and mapping objects. The following methods are available in addition to the standard methods.

`dbhash.first`()¶

It's possible to loop over every key/value pair in the database using this method and the `next()` method. The traversal is ordered by the databases internal hash values, and won't be sorted by the key values. This method returns the starting key.

`dbhash.last`()¶

Return the last key/value pair in a database traversal. This may be used to begin a reverse-order traversal; see `previous()`.

`dbhash.next`()¶

Returns the key next key/value pair in a database traversal. The following code prints every key in the database db, without having to create a list in memory that contains them all:

```
print db.first()
for i in xrange(1, len(db)):
   print db.next()
```

`dbhash.previous`()¶

Returns the previous key/value pair in a forward-traversal of the database. In conjunction with `last()`, this may be used to implement a reverse-order traversal.

`dbhash.sync()`¶

This method forces any unwritten data to be written to the disk.

© Copyright 1990-2010, Python Software Foundation.
The Python Software Foundation is a non-profit corporation. Please donate.
Last updated on Feb 26, 2010. Created using Sphinx 0.6.3.