Fsdb Documentation

Release 1.2.2

ael

March 26, 2016

Contents

CHAPTER 1

About

Fsdb is a python implementation of a content addressable storage, it is designed to work with a huge number of big files and it will use your filesystem in a smart way.

Fsdb is the right library for every one that doesn't want to store big files on his database.

Fsdb will works alongside your favorite database, it will help you to easily store and manage files while your database will handle metadata managment.

CHAPTER 2

Quick start

2.1 Installation

Fsdb is available on PyPI so you can easily install through pip

```
pip install Fsdb
```

2.2 Usage

```
from fsdb import Fsdb

#create new fsdb instance
myFsdb = Fsdb("/tmp/fsdbRoot")

#add file
file_digest = myFsdb.add("/path/to/an/existing/file")

#control if file exists
if file_digest in myFsdb:
    # file exists

#get file object
myFsdb[file_digest]

#get file path
myFsdb.get_file_path(file_digest)

#check file integrity
myFsdb.check(file_digest)

#remove file
myFsdb.remove(file_digest)
```

CHAPTER 3

Configuration

There are two ways to configure fsbd:

- passing arguments to class constructor `Fsdb.__init__()`
- editing the json config file

The config file must be in the fsdb root folder with name `` `.fsdb.conf` `` and must be written in a valid json syntax

config name	type	default value	description
depth	int	3	number of levels to use for directory tree
hash_alg	string	"sha1"	name of the hash algorithm to use for file digest
fmode	string	"660"	permissions mask to use in files creation
dmode	string	see *dmode*	permissions mask to use in folders creation

3.1 dmode

If dmode is not provided, the default value will be used. The default value for dmode will be calculated from the fmode, It will inherit all permissions from fmode and for every role that has read permission will be setted also the execute permission.

CHAPTER 4

Path example

Important: you shouldn't make any assumption about fsdb paths structure. The following explanation is for illustrative purpose only.

If you add a file with the following sha1sum to an fsdb instance with a configured depth level of 3

```
7bf770901365d4b12ce46a2d545407daf224e583
```

The file will be placed in

```
/path_To_Fsdb_Root/7b/f770/901365d4/b12ce46a2d545407daf224e583
```

CHAPTER 5

Api

5.1 fsdb.Fsdb

class fsdb.**Fsdb**(*fsdbRoot*, *depth=None*, *hash_alg=None*, *fmode=None*, *dmode=None*)

File system database expose a simple api (add,get,remove) to menage the saving of files on disk. files are placed under specified fsdb root folder and are managed using a directory tree generated from the file digest

BLOCK_SIZE = 1048576

CONFIG_FILE = u'.fsdb.conf'

_calc_digest(*origin*)

calculate digest for the given file or readable/seekable object

Args: origin – could be the path of a file or a readable/seekable object (fileobject, stream, stringIO...)

Returns: String rapresenting the digest for the given origin

_copy_content(*origin*, *dstPath*)

copy the content of origin into dstPath

Due to concurrency problem, the content will be first copied to a temporary file alongside *dstPath* and then atomically moved to *dstPath*

_create_empty_file(*path*)

_makedirs(*path*)

Make folders recursively for the given path and check read and write permission on the path

Args: path – path to the leaf folder

add(*origin*)

Add new element to fsdb.

Args: origin – could be the path of a file or a readable/seekable object (fileobject, stream, stringIO...)

Returns: String rapresenting the digest of the file

check(*digest*)

Check the integrity of the file with the given digest

Args: digest – digest of the file to check

Returns: True if the file is not corrupted

static config_exists(*fsdbRoot*)

corrupted()
Iterate over digests of all corrupted stored files

exists(*digest*)
Check file existence in fsdb

Returns: True if file exists under this instance of fsdb, false otherwise

static generate_tree_path(*fileDigest*, *depth*)

Generate a relative path from the given fileDigest relative path has a numbers of directories levels according to @depth

Args: fileDigest – digest for which the relative path will be generate depth – number of levels to use in relative path generation

Returns: relative path for the given digest

get_file_path(*digest*)
Retrieve the absolute path to the file with the given digest

Args: digest – digest of the file

Returns: String rapresenting the absolute path of the file

remove(*digest*)

Remove an existing file from fsdb. File with the given digest will be removed from fsdb and the directory tree will be cleaned (remove empty folders)

Args: digest – digest of the file to remove

size()
Return the total size in bytes of all the files handled by this instance of fsdb.

Fsdb does not use auxiliary data structure, so this function could be expensive. Look at _iter_over_paths() functions for more details.

CHAPTER 6

Indices and tables

- genindex
- modindex
- search

Symbols

A

B

C

E

F

G

R

S