Dantalian Documentation

Release 1.0

Allen Li

March 17, 2015

Contents

> *"I ask of thee, art thou mankind?"*
> *"Nay, I am the world, the world inside the gourd."*
> *— The Mystic Archives of Dantalian*

Website http://darkfeline.github.io/dantalian/

This is the documentation for Dantalian 1.0, built on March 17, 2015.

Dantalian is a Python 3 library for file organization and tagging using hard links.

CHAPTER 1

Contents

1.1 Introduction

Dantalian is a Python 3 library that provides convenient functions for file organization and tagging using hard links.

Dantalian is extremely flexible, imposing no constraints on usage; read the documentation carefully so you understand what you are doing.

Dantalian can also be used as a standalone program. However, using Dantalian separately may be slow due to Python's nature. When performing bulk operations, consider using Dantalian as a library in a Python script instead of invoking Dantalian repeatedly in a shell script.

1.2 Installation

Install using packages from your distribution if available. Otherwise, see below for manual installation.

1.2.1 Arch Linux

- dantalian
- dantalian-git

1.2.2 Manual installation

Dependencies:

- Python 3

Build dependencies:

- setuptools
- Sphinx (for documentation)

Installation is simple. Obtain the sources, then run:

```
$ python setup.py install
```

This will most likely require root, and will install Dantalian globally on the system. Otherwise, you can use virtualenv, or install it for the user:

```
$ python setup.py install --user
```

It is recommended to install the man pages as well. The man pages can be built like so:

```
$ cd doc
$ make man
```

The man pages can be found in `doc/_build/man`. How they are installed depends on your system. On Arch Linux, man pages are installed in `/usr/share/man` as gzipped archives, so you would do the following:

```
$ cd doc/_build/man
$ gzip ./*
# install ./* /usr/share/man/man1
```

1.3 Definitions

This section contains definitions of terms used in the documentation.

1.3.1 General

The following are general terms you should be familiar with, but are provided here for clarification and refernce.

pathname, path A string, consisting of filenames separated with forward slashes.

basename The part of a path after the last forward slash in it. If the path ends in a forward slash, then the basename is the empty string.

dirname The part of a path before the last forward slash in it.

filename A string, which in a directory maps to a link. Cannot contain forward slashes. Filenames are components of paths.

hard link, link A directory entry pointing to a file.

file A file in the file system, consisting of its inode and corresponding data blocks. A file has at least one link pointing to it.

directory A special type of file, which maps filenames to links and can only have one link referring to it.

symbolic link, symlink A special type of link, which contains a string instead of pointing to a file. The string is used as a pathname.

1.3.2 Dantalian-specific

The following are terms that are used by Dantalian internally and in this documentation.

tagname A special type of pathname which begins with at least two forward slashes. After stripping all forward slashes from the beginning of a tagname, the remaining string is considered a pathname relative to a given rootpath. See *Tagnames*.

rootpath A pathname that is used to resolve a tagname. See *Tagnames*.

library A directory which contains a link to a directory with the filename *.dantalian*. See *Libraries*.

1.4 Basic linking

This section describes Dantalian's basic linking features. Dantalian's fundamental linking functionality is contained in `dantalian.base`. The main functions defined in this module are `link()`, `unlink()`, and `rename()`, which are analogous to their counterparts in the standard `os` module, except that they have been extended to work with directories (see *Directory linking*). `dantalian.base` additionally includes helper functions to compensate for the implementation of these extended features.

`dantalian.base.`**`link`**`(`*rootpath*, *src*, *dst*`)`

Link *src* to *dst*. See *Directory linking* for how directories are linked.

Parameters

- **rootpath** (*str*) – Path for tagname conversions.
- **src** (*str*) – Source path.
- **dst** (*str*) – Destination path.

`dantalian.base.`**`unlink`**`(`*rootpath*, *path*`)`

Unlink the given path. See *Directory linking* for how directories are unlinked.

If the directory does not have any extra links, `IsADirectoryError` is raised.

If *path* is the actual directory and the directory dies have extra links, the directory is swapped out using `swap_dir()`

Parameters

- **rootpath** (*str*) – Path for tagname conversions.
- **path** (*str*) – Target path.

Raises IsADirectoryError Target is a directory without any other links.

Note: This function does not work recursively for directories. For example, unlinking a directory `foo` that contains a link `bar` to another directory will not properly update `bar`'s `.dtags` file.

`dantalian.base.`**`rename`**`(`*rootpath*, *src*, *dst*`)`

Renames the given link. Implemented as and functionally equivalent to:

```
link(rootpath, src, dst)
unlink(rootpath, src)
```

Parameters

- **rootpath** (*str*) – Path for tagname conversions.
- **src** (*str*) – Source path.
- **dst** (*str*) – Destination path.

Note: This will not overwrite files, unlike `os.rename()`.

Note: This function does not work recursively for directories. For example, renaming a directory `foo` that contains a link `bar` to another directory will not properly update `bar`'s `.dtags` file.

The following function is provided for convenience.

`dantalian.base.`**`list_links`**`(`*top*, *path*`)`

Traverse the directory tree, finding all of the links to the target file.

Parameters

- **top** (*str*) – Path of directory to begin search.
- **path** (*str*) – Path of target file.

Returns Generator yielding paths.

Note: This function returns a generator that lazily traverses the file system. Any changes to the file system will affect the generator's execution.

1.4.1 Directory linking

Directory linking is implemented in Dantalian using symlinks and a file named `.dtags` in each tagged directory. Dantalian assumes that the status of symlinks in the file system are consistent with the contents of the `.dtags` files, except for a number of administrative functions.

A directory is linked thus, given a target path *path* and a rootpath *rootpath*: A symlink is created at *path*, whose target is the absolute path to the directory. A tagname is created given *path* and *rootpath*, which is added to the file named `.dtags` in the directory.

Similarly, a directory is unlinked thus, given a target path *path* and a rootpath *rootpath*: The symlink at *path* is removed, and the tagname created given *path* and *rootpath* is removed from the `.dtags` file in the directory. Unlinking a directory that has no such extra links is invalid.

The following function is provided for convenience.

`dantalian.base.`**`swap_dir`**`(`*rootpath*, *path*`)`

Swap a symlink with its target directory. More specifically, given that an actual directory with path `foo` is also linked at `bar`, calling this function on `bar` will move the actual directory to `bar`, creating a symlink at `foo`, and updating the `.dtags` file appropriately.

This is useful when the actual directory, not a symlink, is needed somewhere.

Parameters

- **rootpath** (*str*) – Path for tagname conversions.
- **path** (*str*) – Target path.

Raises ValueError Target is not a symlink to a directory.

The following are administrative functions that do not necessarily assume that symlink state is consistent with `.dtags` state and are used to repair and maintain such state consistency.

`dantalian.base.`**`save_dtags`**`(`*rootpath*, *top*, *dirpath*`)`

Save the current state of symlinks to the target directory in its `.dtags` file, overwriting its current `.dtags` state. The file system search is done recursively from *top*.

This is useful for "committing" file system changes to `.dtags` files.

Parameters

- **rootpath** (*str*) – Path for tagname conversions.
- **top** (*str*) – Path of search directory.
- **dirpath** (*str*) – Path of target directory.

`dantalian.base.`**`load_dtags`**`(`*rootpath*, *dirpath*`)`

Create symlinks according to the directory's `.dtags` file.

This is useful in conjunction with `unload_dtags()` for moving directory trees around without worrying about symlink targets.

Parameters

- **rootpath** (*str*) – Path for tagname conversions.
- **dirpath** (*str*) – Path of target directory.

`dantalian.base.`**`unload_dtags`**`(`*rootpath*, *dirpath*`)`

Remove symlinks according to the directory's `.dtags` file.

This is useful in conjunction with `load_dtags()` for moving directory trees around without worrying about symlink targets.

Parameters

- **rootpath** (*str*) – Path for tagname conversions.
- **dirpath** (*str*) – Path of target directory.

1.4.2 Tagnames

Tagnames are a special type of pathnames used by Dantalian internally. They begin with at least two forward slashes. After stripping all forward slashes from the beginning of a tagname, the remaining string is considered a pathname relative to a given rootpath.

Tagnames are used in `.dtags` files for tagging directories, as well as as shortcuts for the standalone script.

`dantalian.tagnames` contains functions for working with tagnames. Even though the transformation between tagnames and pathnames is relatively simple, use the functions provided in this module to ensure consistent behavior.

`dantalian.tagnames.`**`is_tag`**`(`*name*`)`

Check if the given path is a tagname.

Parameters **name** (*str*) – Pathname.

Returns Whether the given path is a tagname.

Return type bool

`dantalian.tagnames.`**`path2tag`**`(`*rootpath*, *pathname*`)`

Convert a pathname to a tagname.

This function will also normalize the given path before converting it to a tagname.

Parameters

- **rootpath** (*str*) – Path for tagname conversions.
- **pathname** (*str*) – Pathname.

Returns Tagname.

Return type str

`dantalian.tagnames.`**`tag2path`**`(`*rootpath*, *tagname*`)`

Convert a tagname to a pathname.

This function doesn't normalize the resulting path.

Parameters

- **rootpath** (*str*) – Path for tagname conversions.
- **tagname** (*str*) – Tagname.

Returns Pathname.

Return type str

`dantalian.tagnames.`**`path`**`(`*rootpath*, *name*`)`

Return the given tagname or pathname as a pathname.

In other words, convert the given name to a pathname if it is tagname.

`dantalian.tagnames.`**`tag`**`(`*rootpath*, *name*`)`

Return the given tagname or pathname as a tagname.

In other words, convert the given name to a tagname if it is not a tagname.

1.4.3 Libraries

Libraries are special directories Dantalian uses to make file management more convenient. A library is a directory that contains a subdirectory named `.dantalian`.

Currently, libraries exist to provide a clear *rootpath* to be used by Dantalian's various linking function. The standalone Dantalian script will search parent directories for a library to use as a root for many commands so that you do not have to explicitly provide one yourself. Other scripts using Dantalian as a library can also take advantage of libraries as anchor points.

Currently, `.dantalian` is not used for anything beyond identifying libraries, but in the future, it may be used for caching search results or other caching or data storage purposes.

`dantalian.library` contains functions for working with libraries.

`dantalian.library.`**`is_library`**`(`*dirpath*`)`

Return whether the given directory is a library.

Parameters **dirpath** (*str*) – Path to directory.

Returns Whether directory is library.

Return type bool

`dantalian.library.`**`find_library`**`(`*dirpath='.'*`)`

Find a library. Starting from the given path, search up the file system. Return the path of the first library found, including the initially given path. Returns `None` if no library is found.

Parameters **dirpath** (*str*) – Path to search.

Returns Path or None

Return type str or None

`dantalian.library.`**`init_library`**`(`*dirpath*`)`

Initialize a library. Does nothing if the given directory is already a library.

Parameters **dirpath** (*str*) – Path to directory.

`dantalian.library.`**`get_resource`**`(`*dirpath*, *resource_path*`)`

Get the path of a resource stored in the library.

May be used in the future for library data or cache storage.

1.5 Searching

The `dantalian.findlib` module implements tag queries. See *Tagging* for more information about tags.

Queries are represented as a tree of `SearchNodes`.

Example usage:

```
from dantalian import findlib

# Find files which are tagged foo and bar
paths = findlib.search(findlib.parse_query('AND foo bar END'))
```

`dantalian.findlib.`**`search`**`(`*search_node*`)`

Return a list of result paths for a given search query.

`dantalian.findlib.`**`parse_query`**`(`*rootpath*, *query*`)`

Parse a query string into a query node tree.

Parent node syntax:

```
NODE foo [bar...] END
```

where NODE is AND, OR, or MINUS

Tokens beginning with a backslash are used directly in `DirNode`s. Everything else parses to a :class:`DirNode`.

Tagnames are converted to paths using the given *rootpath*.

Query strings look like:

```
'AND foo bar OR spam eggs END AND \AND \OR \END \\\END END END'
```

which parses to:

```
AndNode([
   DirNode('foo'),
   DirNode('bar'),
   OrNode([
       DirNode('spam'),
       DirNode('eggs'),
   ]),
   AndNode([
       DirNode('AND'),
       DirNode('OR'),
       DirNode('END'),
       DirNode('\\END'),
   ]),
])
```

1.5.1 Query nodes

Query nodes are used to represent a search query. Query node trees can be built manually using the node classes or by using `parse_query()`

class `dantalian.findlib.`**`SearchNode`**

An abstract interface for all query nodes.

get_results(*self*)

Abstract method. Returns the results of query represented by the current node.

Returns A dictionary mapping inode objects to paths.

class `dantalian.findlib.`**GroupNode**(*children*)

Abstract class for query nodes that have a list of child nodes, i.e. non-leaf nodes.

Parameters **children** (*list*) – List of children nodes.

class `dantalian.findlib.`**AndNode**(*children*)

Query node that merges the results of its children nodes by set intersection.

Parameters **children** (*list*) – List of children nodes.

class `dantalian.findlib.`**OrNode**(*children*)

Query node that merges the results of its children nodes by set union.

Parameters **children** (*list*) – List of children nodes.

class `dantalian.findlib.`**MinusNode**(*children*)

Query node that merges the results of its children nodes by set difference: the results of its first child minus the results of the rest of its children.

Parameters **children** (*list*) – List of children nodes.

class `dantalian.findlib.`**DirNode**(*dirpath*)

Query node that returns a directory's contents as results. These are the leaf nodes in a query search tree.

1.6 Tagging

Dantalian provides a simple implementation of tagging with hard links using the module `dantalian.tagging`. Tagging works thusly:

Objects can be arbitrarily tagged with tags. Objects can be both files and directories, and tags can only be directories. An object is tagged with a given tag when it has a link in the corresponding directory. Similarly, an object is untagged by removing all of its links in the corresponding directory.

`dantalian.tagging.`**tag**(*rootpath*, *path*, *directory*)

Tag a file (or directory) with a directory. In effect, this tries to link *path* inside *directory* using `dantalian.base.link()`. It will try to use the same *basename* as the given file if possible; if not, it will try to find a similar name that is free.

`dantalian.tagging.`**untag**(*rootpath*, *path*, *directory*)

Untag a file (or directory) from a directory. Essentially calls `dantalian.base.unlink()` on all links of the target file in the given directory.

1.7 Bulk operations

Bulk operations are defined in `dantalian.bulk`. These functions operate on multiple file or entire directory trees.

`dantalian.bulk.`**clean_symlinks**(*dirpath*)

Remove all broken symlinks in the given directory tree.

`dantalian.bulk.`**rename_all**(*rootpath*, *top*, *path*, *name*)

Rename all links to the given file or directory.

Attempt to rename all links to the target under the rootpath to the given name, finding a name as necessary. If there are multiple links in a directory, the first will be renamed and the rest unlinked.

Parameters

- **rootpath** (*str*) – Base path for tagname conversions.
- **top** (*str*) – Path of search directory.
- **path** (*str*) – Path of target to rename.
- **name** (*str*) – New filename.

`dantalian.bulk.`**`unlink_all`**`(`*rootpath*, *top*, *path*`)`

Unlink all links to the target file or directory. This can be used to completely remove a file instead of needing to manually unlink each of its links.

Parameters

- **rootpath** (*str*) – Base path for tagname conversions.
- **top** (*str*) – Path of search directory.
- **path** (*str*) – Path of target.

1.7.1 Import and export

`dantalian.bulk.`**`import_tags`**`(`*rootpath*, *path_tag_map*`)`

Import a path tag map, such as one returned from `export_tags()`.

Tags each path with the given tagnames, thus “importing” tag data.

Parameters

- **rootpath** (*str*) – Base path for tag conversions.
- **path_tag_map** (*dict*) – Mapping of paths to lists of tagnames.

`dantalian.bulk.`**`export_tags`**`(`*rootpath*, *top*, *full=False*`)`

Export a path tag map.

Each file will only have one key path mapping to a list of tags. If *full* is `True`, each file will have one key path for each one of that file’s links, all mapping to the same list of tags.

Example without *full*:

```
{'foo/file': ['//foo', '//bar']}
```

With `full`:

```
{'foo/file': ['//foo', '//bar'],
 'bar/file': ['//foo', '//bar']}
```

Parameters

- **rootpath** (*str*) – Base path for tag conversions.
- **top** (*str*) – Top of directory tree to export.
- **full** (*bool*) – Whether to include all paths to a file. Defaults to False.

Returns Mapping of paths to lists of tagnames.

Return type dict

1.8 Command reference (man pages)

Documentation for the standalone Dantalian script and its commands are contain in the manual page, which is duplicated and linked below.

1.8.1 dantalian(1) – file management using hard links

SYNOPSIS

dantalian [*options*] *command* [*args*]

DESCRIPTION

dantalian is a standalone script for accessing Dantalian functionality.

Dantalian is a Python 3 library to assist file organization and tagging using hard links.

The commands here are generally equivalent to the respective functions in the Dantalian library, with some command line sugar. Therefore, make sure to read the documentation in addition to the man pages!

OPTIONS

-h, --help Print help information.

COMMANDS

Base commands

dantalian-link(1) Link file or directory.

dantalian-unlink(1) Unlink file or directory.

dantalian-rename(1) Rename file or directory.

dantalian-swap(1) Swap symlink with its directory.

dantalian-save(1) Save dtags.

dantalian-load(1) Load dtags.

dantalian-unload(1) Unoad dtags.

dantalian-list(1) List links.

Search commands

dantalian-search(1) Do tag query search.

Library commands

dantalian-init-library(1) Initialize library.

Tagging commands

dantalian-tag(1) Tag file or directory.

dantalian-untag(1) Untag file or directory.

Bulk commands

dantalian-clean(1) Clean up broken symlinks.

dantalian-rename-all(1) Rename all links of a file.

dantalian-unlink-all(1) Unlink all links of a file.

dantalian-import(1) Import tag data.

dantalian-export(1) Export tag data.

SEE ALSO

Online documentation http://dantalian.readthedocs.org/

Project website http://darkfeline.github.io/dantalian/

1.8.2 dantalian-link(1) – Link file or directory

SYNOPSIS

dantalian link [*options*] *src dst*

DESCRIPTION

Link *src* to *dst*

Replacement for ln(1) that works with directories.

OPTIONS

-h, --help Print help information.

--root=PATH Specify the root directory of the library to use. If not specified, try to find a library automatically.

SEE ALSO

dantalian(1) Main man page

1.8.3 dantalian-unlink(1) – Unlink file or directory

SYNOPSIS

dantalian unlink [*options*] *file...*

DESCRIPTION

Unlink files and/or directories.

Replacement for rm(1) that works with directories.

OPTIONS

-h, --help Print help information.

--root=PATH Specify the root directory of the library to use. If not specified, try to find a library automatically.

SEE ALSO

dantalian(1) Main man page

1.8.4 dantalian-rename(1) – Rename file or directory

SYNOPSIS

dantalian rename [*options*] *src dst*

DESCRIPTION

Rename file or directory.

Replacement for mv(1) that works with directory links.

OPTIONS

-h, --help Print help information.

--root=PATH Specify the root directory of the library to use. If not specified, try to find a library automatically.

SEE ALSO

dantalian(1) Main man page

1.8.5 dantalian-swap(1) – Swap directory links

SYNOPSIS

dantalian swap [*options*] *dir*

DESCRIPTION

Swap out a directory symlink for the actual directory.

OPTIONS

-h, --help Print help information.

--root=PATH Specify the root directory of the library to use. If not specified, try to find a library automatically.

SEE ALSO

dantalian(1) Main man page

1.8.6 dantalian-save(1) – Save dtags

SYNOPSIS

dantalian save [*options*] *dir*

DESCRIPTION

Save file system symlink information to a directory's dtags file.

OPTIONS

-h, --help Print help information.

--root=PATH Specify the root directory of the library to use. If not specified, try to find a library automatically.

--all Recursively save for all directories.

SEE ALSO

dantalian(1) Main man page

1.8.7 dantalian-load(1) – Load dtags

SYNOPSIS

dantalian load [*options*] *dir*

DESCRIPTION

Load file system symlink information from a directory's dtags file.

OPTIONS

-h, --help Print help information.

--root=PATH Specify the root directory of the library to use. If not specified, try to find a library automatically.

--all Recursively load for all directories.

SEE ALSO

dantalian(1) Main man page

1.8.8 dantalian-unload(1) – Unload dtags

SYNOPSIS

dantalian unload [*options*] *dir*

DESCRIPTION

Unload file system symlinks using a directory's dtags file.

OPTIONS

-h, --help Print help information.

--root=PATH Specify the root directory of the library to use. If not specified, try to find a library automatically.

--all Recursively unload for all directories.

SEE ALSO

dantalian(1) Main man page

1.8.9 dantalian-list(1) – List links

SYNOPSIS

dantalian list [*options*] *path*

DESCRIPTION

List all links of the given file or directory.

OPTIONS

-h, --help Print help information.

--root=PATH Specify the root directory of the library to use. If not specified, try to find a library automatically.

--tags List tagnames instead of pathnames.

SEE ALSO

dantalian(1) Main man page

1.8.10 dantalian-search(1) – Search tags

SYNOPSIS

dantalian search [*options*] *query...*

DESCRIPTION

Do a tag query search. Queries are parsed using the Dantalian library; see documentation for details.

OPTIONS

-h, --help Print help information.

--root=PATH Specify the root directory of the library to use. If not specified, try to find a library automatically.

SEE ALSO

dantalian(1) Main man page

1.8.11 dantalian-init-library(1) – Initialize library

SYNOPSIS

dantalian init-library [*options*] [*path*]

DESCRIPTION

Initialize library. Use current directory if no path is given.

OPTIONS

-h, --help Print help information.

SEE ALSO

dantalian(1) Main man page

1.8.12 dantalian-tag(1) – Tag files

SYNOPSIS

dantalian tag [*options*] -f *file*... – *tag*...

DESCRIPTION

Tag all of the given files with all of the given tags.

OPTIONS

-h, --help Print help information.

--root=PATH Specify the root directory of the library to use. If not specified, try to find a library automatically.

SEE ALSO

dantalian(1) Main man page

1.8.13 dantalian-untag(1) – Untag files

SYNOPSIS

dantalian untag [*options*] -f *file*... – *tag*...

DESCRIPTION

Remove all of the given tags from all of the given files.

OPTIONS

-h, --help Print help information.

--root=PATH Specify the root directory of the library to use. If not specified, try to find a library automatically.

SEE ALSO

dantalian(1) Main man page

1.8.14 dantalian-clean(1) – Clean broken symlinks

SYNOPSIS

dantalian clean [*options*] [*dir*]

DESCRIPTION

Remove all broken symlinks. Use current directory if no path is given.

OPTIONS

-h, --help Print help information.

SEE ALSO

dantalian(1) Main man page

1.8.15 dantalian-rename-all(1) – Rename all links

SYNOPSIS

dantalian rename-all [*options*] *path name*

DESCRIPTION

Rename all links of a file or directory.

OPTIONS

-h, --help Print help information.

--root=PATH Specify the root directory of the library to use. If not specified, try to find a library automatically.

SEE ALSO

dantalian(1) Main man page

1.8.16 dantalian-unlink-all(1) – Unlink all links

SYNOPSIS

dantalian unlink-all [*options*] *path...*

DESCRIPTION

Unlink all links of the given files or directories.

OPTIONS

-h, --help Print help information.

--root=PATH Specify the root directory of the library to use. If not specified, try to find a library automatically.

SEE ALSO

dantalian(1) Main man page

1.8.17 dantalian-import(1) – Import tags

SYNOPSIS

dantalian import [*options*]

DESCRIPTION

Import JSON tag data from stdin.

OPTIONS

-h, --help Print help information.

--root=PATH Specify the root directory of the library to use. If not specified, try to find a library automatically.

SEE ALSO

dantalian(1) Main man page

1.8.18 dantalian-export(1) – Export tags

SYNOPSIS

dantalian export [*options*] *dir*

DESCRIPTION

Export JSON tag data to stdout.

OPTIONS

-h, --help Print help information.

--root=PATH Specify the root directory of the library to use. If not specified, try to find a library automatically.

--full Export full tag data; check documentation for more info.

SEE ALSO

dantalian(1) Main man page

CHAPTER 2

Copyright

Python Module Index

Index