python_hosts Documentation

Release 0.3.2

Jon Hadfield

Aug 28, 2023

Contents

A python library for managing a hosts file.

CHAPTER 1

Getting started

1.1 Installation

Using pip package manager

$ pip install python-hosts

From source

$ git clone https://github.com/jonhadfield/python-hosts
$ cd python-hosts
$ python setup.py install

1.2 Usage

Create an instance of a hosts file:

```
from python_hosts import Hosts, HostsEntry
my_hosts = Hosts()
```

Add an entry:

```
new_entry = HostsEntry(entry_type='ipv4', address='1.2.3.4', names=['example.com',
↪'example'])
my_hosts.add([new_entry])
```

Remove an entry/entries matching an address:

```
my_hosts.remove_all_matching(address='1.2.3.4')
```

Remove an entry/entries matching an address:

```
my_hosts.remove_all_matching(name='example.com')
```

Write entries:

```
my_hosts.write()
```

CHAPTER 2

API

2.1 Package

This package contains all the modules utilised by the python-hosts library.

hosts: Contains the Hosts and HostsEntry classes that represent instances of a hosts file, and it's individual lines/entries

utils: Contains helper functions to check the available operations on a hosts file and the validity of a hosts file entry

exception: Contains the custom exceptions that are raised in the event of an error in processing a hosts file and its entries

2.1.1 Submodules

python_hosts.hosts module

This module contains classes: HostsEntry: A representation of a hosts file entry, i.e. a line containing an IP address and name(s), a comment, or a blank line/line separator.

Hosts: A representation of a hosts file, e.g. /etc/hosts and c:\windows\system32\drivers\etc\hosts for a linux or MS windows based machine respectively. Each entry being represented as an instance of the HostsEntry class.

class python_hosts.hosts.**Hosts**(*path=None*, *entries=None*)
Bases: object

A hosts file.

add(*entries=None*, *force=False*, *allow_address_duplication=False*, *merge_names=False*)
Add instances of HostsEntry to the instance of Hosts. :param entries: A list of instances of HostsEntry :param force: Remove matching before adding :param allow_address_duplication: Allow using multiple entries

for same address

Parameters **merge_names** – Merge names where address already exists

Returns The counts of successes and failures

count()
Get a count of the number of host entries :return: The number of host entries

static determine_hosts_path(*platform=None*)
Return the hosts file path based on the supplied or detected platform. :param platform: a string used to identify the platform :return: detected filesystem path of the hosts file

entries

exists(*address=None*, *names=None*, *comment=None*)

Determine if the supplied address and/or names, or comment, exists in a HostsEntry within Hosts

Parameters

- **address** – An ipv4 or ipv6 address to search for
- **names** – A list of names to search for
- **comment** – A comment to search for

Returns True if a supplied address, name, or comment is found. Otherwise, False.

find_all_matching(*address=None*, *name=None*, *comment=None*)

Return all HostsEntry instances from the Hosts object where the supplied ip address or name matches

Parameters

- **address** – An ipv4 or ipv6 address
- **name** – A host name
- **comment** – A host inline comment

Returns HostEntry instances

static get_hosts_by_url(*url=None*)
Request the content of a URL and return the response :param url: The URL of the hosts file to download :return: The content of the passed URL

import_file(*import_file_path=None*)
Read a list of host entries from a file, convert them into instances of HostsEntry and then append to the list of entries in Hosts :param import_file_path: The path to the file containing the host entries :return: Counts reflecting the attempted additions

import_url(*url=None*, *force=None*)

Read a list of host entries from a URL, convert them into instances of HostsEntry and then append to the list of entries in Hosts

Parameters **url** – The URL of where to download a hosts file

Returns Counts reflecting the attempted additions

path

populate_entries()

Called by the initialiser of Hosts. This reads the entries from the local hosts file, converts them into instances of HostsEntry and adds them to the Hosts list of entries.

Returns None

remove_all_matching(*address=None*, *name=None*, *comment=None*)
Remove all HostsEntry instances from the Hosts object where the supplied ip address, name or comment matches :param address: An ipv4 or ipv6 address :param name: A host name :param comment: A host inline comment :return: None

write(*path=None*, *mode='w'*)
Write all of the HostsEntry instances back to the hosts file :param path: override the write path :return: Dictionary containing counts

class python_hosts.hosts.**HostsEntry**(*entry_type=None*, *address=None*, *comment=None*, *names=None*)
Bases: object

An entry in a hosts file.

address

comment

entry_type

static get_entry_type(*hosts_entry=None*)
Return the type of entry for the line of hosts file passed :param hosts_entry: A line from the hosts file :return: 'comment' | 'blank' | 'ipv4' | 'ipv6'

is_real_entry()

names

static str_to_hostentry(*entry*)
Transform a line from a hosts file into an instance of HostsEntry :param entry: A line from the hosts file :return: An instance of HostsEntry

python_hosts.utils module

This module contains utility functions used by the Hosts and HostsEntry methods

python_hosts.utils.**dedupe_list**(*seq*)
Utility function to remove duplicates from a list :param seq: The sequence (list) to deduplicate :return: A list with original duplicates removed

python_hosts.utils.**is_ipv4**(*entry*)
Check if the string provided is a valid ipv4 address :param entry: A string representation of an IP address :return: True if valid, False if invalid

python_hosts.utils.**is_ipv6**(*entry*)
Check if the string provided is a valid ipv6 address :param entry: A string representation of an IP address :return: True if valid, False if invalid

python_hosts.utils.**is_readable**(*path=None*)
Test if the supplied filesystem path can be read :param path: A filesystem path :return: True if the path is a file that can be read. Otherwise, False

python_hosts.utils.**valid_hostnames**(*hostname_list*)
Check if the supplied list of strings are valid hostnames :param hostname_list: A list of strings :return: True if the strings are valid hostnames, False if not

python_hosts.exception module

hosts.exceptions

All exceptions used in the hosts code base are defined here.

exception python_hosts.exception.**HostsEntryException**
Bases: exceptions.Exception

Base exception class. All HostsEntry-specific exceptions should subclass this class.

exception python_hosts.exception.**HostsException**
Bases: exceptions.Exception

Base exception class. All Hosts-specific exceptions should subclass this class.

exception python_hosts.exception.**InvalidComment**
Bases: *python_hosts.exception.HostsEntryException*

Raised when a HostsEntry is defined as type 'comment' but with an invalid comment

exception python_hosts.exception.**InvalidIPv4Address**
Bases: *python_hosts.exception.HostsEntryException*

Raised when a HostsEntry is defined as type 'ipv4' but with an invalid address.

exception python_hosts.exception.**InvalidIPv6Address**
Bases: *python_hosts.exception.HostsEntryException*

Raised when a HostsEntry is defined as type 'ipv6' but with an invalid address.

exception python_hosts.exception.**UnableToWriteHosts**
Bases: *python_hosts.exception.HostsException*

Raised when a Hosts file cannot be written.

2.2 Indices and tables

- genindex
- modindex
- search

Python Module Index

p

Index