
mit_maira Documentation

Release 0.0.4

MIT ODL

May 17, 2018

Contents

| | | |
|----------|----------------------------|----------|
| 1 | API | 3 |
| | Python Module Index | 7 |

Python client for accessing MIT's [Moira](#) system. This client uses the [SOAP](#) API, which has a few unusual limitations, and requires X.509 client certificates for access.

CHAPTER 1

API

class `mit_moire.Moira` (*cert_path*, *key_path*, *proxy_id*=", *url*='https://moiraws.mit.edu/moiraws/services/moira?wsdl')

The client that accesses Moira's SOAP API, powered by `zeep`. Requires an X.509 certificate and private key.

Parameters

- **cert_path** (*str*) – The path to an X.509 certificate file
- **key_path** (*str*) – The path to an X.509 private key file
- **proxy_id** (*str*) – Used in many API calls. Do not set this unless you know what you're doing.
- **url** (*str*) – The WSDL URL to connect to.

client = `None`

An instance of `zeep.Client`

user_lists (*username*, *member_type*='USER')

Look up all the lists that the user is a member of.

Parameters

- **username** (*str*) – The MIT username of the user
- **member_type** (*str*) – The type of user, "USER" or "STRING"

Returns names of the lists that this user is a member of

Return type list of strings

user_list_membership (*username*, *member_type*='USER', *recursive*=True, *max_return_count*=999)

Get info for lists a user is a member of.

This is similar to `user_lists()` but with a few differences:

1. It returns list info objects instead of list names.
2. It has an option to fully resolve a user's list hierarchy. That is, if a user is a member of a nested list, this method can retrieve both the nested list and the parent lists that contain the nested list.

Parameters

- **username** (*str*) – The MIT username of the user
- **member_type** (*str*) – The type of user, “USER” or “STRING”
- **recursive** (*bool*) – Whether to fully resolve the list hierarchy
- **max_return_count** (*int*) – limit the number of items returned

Returns info dicts, one per list.

Return type list of dicts

list_members (*name*, *type*=‘USER’, *recurse*=True, *max_results*=1000)

Look up all the members of a list.

Parameters

- **name** (*str*) – The name of the list
- **type** (*str*) – The type of results to return. “USER” to get users, “LIST” to get lists.
- **recurse** (*bool*) – Presumably, whether to recurse into member lists when retrieving users.
- **max_results** (*int*) – Maximum number of results to return.

Returns names of the members of the list

Return type list of strings

list_attributes (*name*)

Look up the attributes of a list.

Parameters **name** (*str*) – The name of the list

Returns attributes of the list

Return type dict

list_exists (*name*)

Does this list exist?

Parameters **name** (*str*) – The name of the list

Returns whether the list exists

Return type bool

add_member_to_list (*username*, *listname*, *member_type*=‘USER’)

Add a member to an existing list.

Parameters

- **username** (*str*) – The username of the user to add
- **listname** (*str*) – The name of the list to add the user to
- **member_type** (*str*) – Normally, this should be “USER”. If you are adding a list as a member of another list, set this to “LIST”, instead.

create_list (*name*, *description*=‘Created by mit_moira client’, *is_active*=True, *is_public*=True, *is_hidden*=True, *is_group*=False, *is_nfs_group*=False, *is_mail_list*=False, *use_mailman*=False, *mailman_server*=‘’)

Create a new list.

Parameters

- **name** (*str*) – The name of the new list
- **description** (*str*) – A short description of this list
- **is_active** (*bool*) – Should the new list be active? An inactive list cannot be used.
- **is_public** (*bool*) – Should the new list be public? If a list is public, anyone may join without requesting permission. If not, the owners control entry to the list.
- **is_hidden** (*bool*) – Should the new list be hidden? Presumably, a hidden list doesn't show up in search queries.
- **is_group** (*bool*) – Something about AFS?
- **is_nfs_group** (*bool*) – Presumably, create an [NFS group](#) for this group? I don't actually know what this does.
- **is_mail_list** (*bool*) – Presumably, create a mailing list.
- **use_mailman** (*bool*) – Presumably, use [GNU Mailman](#) to manage the mailing list.
- **mailman_server** (*str*) – The Mailman server to use, if `use_mailman` is True.

update_list (*name*, *new_name=None*, *description='Updated by mit_maira client'*, *is_active=True*, *is_public=True*, *is_hidden=True*, *is_group=False*, *is_nfs_group=False*, *is_mail_list=False*, *use_mailman=False*, *mailman_server=""*)

Update an existing list. Be warned that this will overwrite *all* attributes on the list, not just the ones you specify!

Parameters

- **name** (*str*) – The name of the existing list to be updated
- **new_name** (*str*) – If you wish to change the name of the list, set it here. Otherwise, the name will remain the same.
- **description** (*str*) – A short description of this list
- **is_active** (*bool*) – Should the list be active? An inactive list cannot be used.
- **is_public** (*bool*) – Should the list be public? If a list is public, anyone may join without requesting permission. If not, the owners control entry to the list.
- **is_hidden** (*bool*) – Should the list be hidden? Presumably, a hidden list doesn't show up in search queries.
- **is_group** (*bool*) – Something about AFS?
- **is_nfs_group** (*bool*) – Presumably, have an [NFS group](#) for this group? I don't actually know what this does.
- **is_mail_list** (*bool*) – Presumably, have a mailing list.
- **use_mailman** (*bool*) – Presumably, have [GNU Mailman](#) manage the mailing list.
- **mailman_server** (*str*) – The Mailman server to use, if `use_mailman` is True.

print_capabilities ()

Print out the capabilities of this SOAP API.

m

`mit_maira`, 3

A

`add_member_to_list()` (`mit_maira.Moira` method), 4

C

`client` (`mit_maira.Moira` attribute), 3

`create_list()` (`mit_maira.Moira` method), 4

L

`list_attributes()` (`mit_maira.Moira` method), 4

`list_exists()` (`mit_maira.Moira` method), 4

`list_members()` (`mit_maira.Moira` method), 4

M

`mit_maira` (module), 1

`Moira` (class in `mit_maira`), 3

P

`print_capabilities()` (`mit_maira.Moira` method), 5

U

`update_list()` (`mit_maira.Moira` method), 5

`user_list_membership()` (`mit_maira.Moira` method), 3

`user_lists()` (`mit_maira.Moira` method), 3