PyLmod Documentation

Release 0.2.0

MIT Office of Digital Learning

Nov 27, 2018

Contents

1 Getting Started **3**

2 Licensing **5**

3 Table of Contents **7**

3.1 PyLmod API Docs 7

3.1.1 Base Class 7

3.1.2 Gradebook Class 9

3.1.3 Membership Class 21

4 Indices and Search **25**

Python Module Index **27**

PyLmod Python implementation of MIT Learning Modules API

Version 0.2.0

Author MIT Office of Digital Learning

Homepage http://engineering.odl.mit.edu

License BSD

PyLmod provides a Python library to access the MIT Learning Modules web service (described below). PyLmod was created to support MIT's use of OpenedX for residential courses, but the library is open source to enable easier access to that service for Python application developers at MIT. PyLmod encapsulates the Learning Modules web service making it more pythonic and easier to incorporate into Python applications.

The MIT Learning Modules web service, maintained by MIT Information Systems and Technologies (IS&T), exposes an API to MIT systems of record for classes, students, and grades. Its documentation is available at these links.

MIT Learning Modules web service documentation:

Gradebook module doc

Membership module doc

CHAPTER 1

Getting Started

The Learning Modules web service requires authentication by x.509 certificates. You must create an application certificate and configure the Learning Modules web service to recognize it. MIT developers can use this IS&T guide to create an application certificate. The MITx Knowledge Base also contains an article 'MIT Application Certificates" that explains the steps in greater detail.

Once you have your application certificate you must get the Learning Modules service to recognize it. The app certificate needs to have an account on the service and then the proper role(s) in the proper group(s). Send your application certificate to learningmod-support@mit.edu with a request for access. Inform them what your application will do and they will assist in configuring your certificate. This service, maintained by MIT Information Systems and Technologies (IS&T) exposes an API to MIT systems of record for classes, students, and grades. PyLmod was created to support MIT's use of OpenedX for residential courses, but the library is open source to enable easier access for Python application developers at MIT.

CHAPTER 2

Licensing

PyLmod is licensed under the BSD license, version January 9, 2008. See LICENSE for the full text of the license.

CHAPTER 3

Table of Contents

3.1 PyLmod API Docs

For convenient reference in development, here are the PyLmod API docs.

3.1.1 Base Class

Python class representing interface to MIT Learning Modules Web service.

class pylmod.base.**Base**(*cert*, *urlbase='https://learning-modules.mit.edu:8443/'*)

Bases: object

Base provides the transport for accessing the MIT Learning Modules (LMod).

The Base class implements the functions that underlie the HTTP calls to the MIT Learning Modules (LMod) Web service. It shouldn't be instantiated directly as it is inherited by the classes that implement the API.

cert

File path to the certificate used to authenticate access to LMod Web service

Type unicode

urlbase

The URL of the LMod Web service. i.e. learning-modules.mit.edu or learning-modules-test.mit.edu

Type str

Initialize Base instance.

Parameters

- **cert** (*unicode*) – File path to the certificate used to authenticate access to LMod Web service
- **urlbase** (*str*) – The URL of the LMod Web service. i.e. learning-modules.mit.edu or learning-modules-test.mit.edu

RETRIES = 10

TIMEOUT = 200

delete(*service*)
Generic DELETE operation for Learning Modules API.

Parameters **service** (*str*) – The endpoint service to use, i.e. gradebook

Raises

- **requests.RequestException** – Exception connection error
- **ValueError** – Unable to decode response content

Returns the json-encoded content of the response

Return type list

get(*service*, *params=None*)
Generic GET operation for retrieving data from Learning Modules API.

```
gbk.get('students/{gradebookId}', params=params, gradebookId=gbid)
```

Parameters

- **service** (*str*) – The endpoint service to use, i.e. gradebook
- **params** (*dict*) – additional parameters to add to the call

Raises

- **requests.RequestException** – Exception connection error
- **ValueError** – Unable to decode response content

Returns the json-encoded content of the response

Return type list

gradebookid = None

post(*service*, *data*)
Generic POST operation for sending data to Learning Modules API.

Data should be a JSON string or a dict. If it is not a string, it is turned into a JSON string for the POST body.

Parameters

- **service** (*str*) – The endpoint service to use, i.e. gradebook
- **data** (*json or dict*) – the data payload

Raises

- **requests.RequestException** – Exception connection error
- **ValueError** – Unable to decode response content

Returns the json-encoded content of the response

Return type list

rest_action(*func*, *url*, ***kwargs*)
Routine to do low-level REST operation, with retry.

Parameters

- **func** (*callable*) – API function to call
- **url** (*str*) – service URL endpoint
- **kwargs** (*dict*) – addition parameters

Raises

- **requests.RequestException** – Exception connection error
- **ValueError** – Unable to decode response content

Returns the json-encoded content of the response

Return type list

verbose = True

3.1.2 Gradebook Class

Contains GradeBook class

class pylmod.gradebook.**GradeBook**(*cert*, *urlbase='https://learning-modules.mit.edu:8443/'*, *gbuuid=None*)

Bases: *pylmod.base.Base*

Since the MIT Learning Modules Web service (LMod) usually returns response data, GradeBook API calls will return this data as a Python data structure, either a list or a dictionary. The data structure will often contain these items:

- `status` - `1` = successful, `-1` = failed
- `message` - details about any error condition, or success message
- `data` - the returned data in a list or dictionary, if applicable

For example, errors usually return a dictionary containing `status` and `message`.

The response is in this format:

```
{
    u'status':1,
    u'message':u'',
    u'data':{...}
}
```

API reference at https://learning-modules-dev.mit.edu/service/gradebook/doc.html

create_assignment(*name*, *short_name*, *weight*, *max_points*, *due_date_str*, *gradebook_id=''*, ***kwargs*)

Create a new assignment.

Create a new assignment. By default, assignments are created under the *Uncategorized* category.

Parameters

- **name** (*str*) – descriptive assignment name, i.e. `new NUMERIC SIMPLE ASSIGNMENT`
- **short_name** (*str*) – short name of assignment, one word of no more than 5 characters, i.e. `SAnew`
- **weight** (*str*) – floating point value for weight, i.e. `1.0`
- **max_points** (*str*) – floating point value for maximum point total, i.e. `100.0`

- **due_date_str** (*str*) – due date as string in `mm-dd-yyyy` format, i.e. `08-21-2011`
- **gradebook_id** (*str*) – unique identifier for gradebook, i.e. `2314`
- **kwargs** (*dict*) – dictionary containing additional parameters, i.e. `graderVisible`, `totalAverage`, and `categoryId`.

  For example:

```
{
    u'graderVisible': True,
    u'totalAverage': None
    u'categoryId': 1007964,
}
```

Raises

- **requests.RequestException** – Exception connection error
- **ValueError** – Unable to decode response content

Returns

dictionary containing data, status and message for example:

```
{
    u'data':
    {
        u'assignmentId': 18490492,
        u'categoryId': 1293820,
        u'description': u'',
        u'dueDate': 1312171200000,
        u'dueDateString': u'08-01-2011',
        u'gradebookId': 1293808,
        u'graderVisible': False,
        u'gradingSchemeId': 18490493,
        u'gradingSchemeType': u'NUMERIC',
        u'isComposite': False,
        u'isHomework': False,
        u'maxPointsTotal': 100.0,
        u'name': u'new NUMERIC SIMPLE ASSIGNMENT',
        u'numStudentGradesToBeApproved': 0,
        u'numStudentsToBeGraded': 614,
        u'shortName': u'SAnew',
        u'userDeleted': False,
        u'weight': 1.0
    },
    u'message': u'assignment is created successfully',
    u'status': 1
}
```

Return type dict

delete_assignment(*assignment_id*)

Delete assignment.

Delete assignment specified by assignment Id.

Parameters **assignment_id** (*str*) – id of assignment to delete

Raises

- **requests.RequestException** – Exception connection error

- **ValueError** – Unable to decode response content

Returns

dictionary containing response `status` and `message`

```
{
    u'message': u'assignment is deleted successfully',
    u'status': 1
}
```

Return type dict

get_assignment_by_name(*assignment_name*, *assignments=None*)

Get assignment by name.

Get an assignment by name. It works by retrieving all assignments and returning the first assignment with a matching name. If the optional parameter `assignments` is provided, it uses this collection rather than retrieving all assignments from the service.

Parameters

- **assignment_name** (*str*) – name of assignment
- **assignments** (*list*) – assignments to search, default: None When `assignments` is unspecified, all assignments are retrieved from the service.

Raises

- **requests.RequestException** – Exception connection error
- **ValueError** – Unable to decode response content

Returns

tuple of assignment id and assignment dictionary

```
(
    16708850,
    {
        u'assignmentId': 16708850,
        u'categoryId': 1293820,
        u'description': u'',
        u'dueDate': 1383541200000,
        u'dueDateString': u'11-04-2013',
        u'gradebookId': 1293808,
        u'graderVisible': False,
        u'gradingSchemeId': 16708851,
        u'gradingSchemeType': u'NUMERIC',
        u'isComposite': False,
        u'isHomework': False,
        u'maxPointsTotal': 100.0,
        u'name': u'midterm1',
        u'shortName': u'mid1',
        u'userDeleted': False,
        u'weight': 1.0
    }
)
```

Return type tuple

get_assignments(*gradebook_id=''*, *simple=False*, *max_points=True*, *avg_stats=False*, *grading_stats=False*)

Get assignments for a gradebook.

Return list of assignments for a given gradebook, specified by a py:attribute::gradebook_id. You can control if additional parameters are returned, but the response time with py:attribute::avg_stats and py:attribute::grading_stats enabled is significantly longer.

Parameters

- **gradebook_id** (*str*) – unique identifier for gradebook, i.e. `2314`
- **simple** (*bool*) – return just assignment names, default= `False`
- **max_points** (*bool*) – Max points is a property of the grading scheme for the assignment rather than a property of the assignment itself, default= `True`
- **avg_stats** (*bool*) – return average grade, default= `False`
- **grading_stats** (*bool*) – return grading statistics, i.e. number of approved grades, unapproved grades, etc., default= `False`

Raises

- **requests.RequestException** – Exception connection error
- **ValueError** – Unable to decode response content

Returns

list of assignment dictionaries

An example return value is:

```
[
    {
        u'assignmentId': 2431240,
        u'categoryId': 1293820,
        u'description': u'',
        u'dueDate': 1372392000000,
        u'dueDateString': u'06-28-2013',
        u'gradebookId': 1293808,
        u'graderVisible': True,
        u'gradingSchemeId': 2431243,
        u'gradingSchemeType': u'NUMERIC',
        u'isComposite': False,
        u'isHomework': False,
        u'maxPointsTotal': 10.0,
        u'name': u'Homework 1',
        u'shortName': u'HW1',
        u'userDeleted': False,
        u'weight': 1.0
    },
    {
        u'assignmentId': 16708850,
        u'categoryId': 1293820,
        u'description': u'',
        u'dueDate': 1383541200000,
        u'dueDateString': u'11-04-2013',
        u'gradebookId': 1293808,
        u'graderVisible': False,
        u'gradingSchemeId': 16708851,
```

(continues on next page)

(continued from previous page)

```
        u'gradingSchemeType': u'NUMERIC',
        u'isComposite': False,
        u'isHomework': False,
        u'maxPointsTotal': 100.0,
        u'name': u'midterm1',
        u'shortName': u'mid1',
        u'userDeleted': False,
        u'weight': 1.0
    },
]
```

Return type list

get_gradebook_id(*gbuuid*)

Return gradebookid for a given gradebook uuid.

Parameters **gbuuid** (*str*) – gradebook uuid, i.e. `STELLAR:/project/gbngtest`

Raises

- **PyLmodUnexpectedData** – No gradebook id returned
- **requests.RequestException** – Exception connection error
- **ValueError** – Unable to decode response content

Returns value of gradebook id

Return type str

get_options(*gradebook_id*)

Get options for gradebook.

Get options dictionary for a gradebook. Options include gradebook attributes.

Parameters **gradebook_id** (*str*) – unique identifier for gradebook, i.e. `2314`

Returns

```
{
    u'data':
    {
        u'accessLevel': u'class',
        u'archived': False,
        u'calc_on_approved_only': False,
        u'configured': None,
        u'courseName': u'',
        u'courseNumber': u'mitxdemosite',
        u'deriveOverallGrades': False,
        u'gradebookEwsEnabled': False,
        u'gradebookId': 1293808,
        u'gradebookName': u'Gradebook for mitxdemosite',
        u'gradebookReadOnly': False,
        u'gradebookVisibleToAdvisors': False,
        u'graders_change_approved': False,
        u'hideExcuseButtonInUI': False,
        u'homeworkBetaEnabled': False,
        u'membershipQualifier': u'/project/mitxdemosite',
        u'membershipSource': u'stellar',
        u'student_sees_actual_grades': True,
        u'student_sees_category_info': True,
```

(continues on next page)

(continued from previous page)

```
        u'student_sees_comments': True,
        u'student_sees_cumulative_score': True,
        u'student_sees_histograms': True,
        u'student_sees_submissions': False,
        u'ta_approves': False,
        u'ta_change_approved': False,
        u'ta_configures': False,
        u'ta_edits': False,
        u'use_grade_weighting': False,
        u'usingAttendance': False,
        u'versionCompatible': 4,
        u'versionCompatibleString': u'General Availability'
    },
}
```

Return type An example return value is

get_section_by_name(*section_name*)

Get a section by its name.

Get a list of sections for a given gradebook, specified by a gradebookid.

Parameters **section_name** (*str*) – The section's name.

Raises

- **requests.RequestException** – Exception connection error
- **ValueError** – Unable to decode response content

Returns

tuple of group id, and section dictionary

An example return value is:

```
(
    1327565,
    {
        u'editable': True,
        u'groupId': 1327565,
        u'groupingScheme': u'Recitation',
        u'members': None,
        u'name': u'r01',
        u'shortName': u'r01',
        u'staffs': None
    }
)
```

Return type tuple

get_sections(*gradebook_id=''*, *simple=False*)

Get the sections for a gradebook.

Return a dictionary of types of sections containing a list of that type for a given gradebook. Specified by a gradebookid.

If simple=True, a list of dictionaries is provided for each section regardless of type. The dictionary only contains one key `SectionName`.

Parameters

- **gradebook_id** (*str*) – unique identifier for gradebook, i.e. 2314
- **simple** (*bool*) – return a list of section names only

Raises

- **requests.RequestException** – Exception connection error
- **ValueError** – Unable to decode response content

Returns

Dictionary of section types where each type has a list of sections

An example return value is:

```
{
    u'recitation':
    [
        {
            u'editable': False,
            u'groupId': 1293925,
            u'groupingScheme': u'Recitation',
            u'members': None,
            u'name': u'Unassigned',
            u'shortName': u'DefaultGroupNoCollisionPlease1234',
            u'staffs': None
        },
        {
            u'editable': True,
            u'groupId': 1327565,
            u'groupingScheme': u'Recitation',
            u'members': None,
            u'name': u'r01',
            u'shortName': u'r01',
            u'staffs': None},
            {u'editable': True,
            u'groupId': 1327555,
            u'groupingScheme': u'Recitation',
            u'members': None,
            u'name': u'r02',
            u'shortName': u'r02',
            u'staffs': None
        }
    ]
}
```

Return type dict

get_staff(*gradebook_id*, *simple=False*)

Get staff list for gradebook.

Get staff list for the gradebook specified. Optionally, return a less detailed list by specifying `simple = True`.

If simple=True, return a list of dictionaries, one dictionary for each member. The dictionary contains a member's `email`, `displayName`, and `role`. Members with multiple roles will appear in the list once for each role.

Parameters

- **gradebook_id** (*str*) – unique identifier for gradebook, i.e. 2314

- **simple** (*bool*) – Return a staff list with less detail. Default is `False`.

Returns

```
{
    u'data': {
        u'COURSE_ADMIN': [
            {
                u'accountEmail': u'benfranklin@mit.edu',
                u'displayName': u'Benjamin Franklin',
                u'editable': False,
                u'email': u'benfranklin@mit.edu',
                u'givenName': u'Benjamin',
                u'middleName': None,
                u'mitId': u'921344431',
                u'nickName': u'Benjamin',
                u'personId': 10710616,
                u'sortableName': u'Franklin, Benjamin',
                u'surname': u'Franklin',
                u'year': None
            },
        ],
        u'COURSE_PROF': [
            {
                u'accountEmail': u'dduck@mit.edu',
                u'displayName': u'Donald Duck',
                u'editable': False,
                u'email': u'dduck@mit.edu',
                u'givenName': u'Donald',
                u'middleName': None,
                u'mitId': u'916144889',
                u'nickName': u'Donald',
                u'personId': 8117160,
                u'sortableName': u'Duck, Donald',
                u'surname': u'Duck',
                u'year': None
            },
        ],
        u'COURSE_TA': [
            {
                u'accountEmail': u'hduck@mit.edu',
                u'displayName': u'Huey Duck',
                u'editable': False,
                u'email': u'hduck@mit.edu',
                u'givenName': u'Huey',
                u'middleName': None,
                u'mitId': u'920445024',
                u'nickName': u'Huey',
                u'personId': 1299059,
                u'sortableName': u'Duck, Huey',
                u'surname': u'Duck',
                u'year': None
            },
        ]
    },
}
```

Return type An example return value is

get_student_by_email(*email*, *students=None*)
Get a student based on an email address.

Calls `self.get_students()` to get list of all students, if not passed as the `students` parameter.

Parameters

- **email** (*str*) – student email
- **students** (*list*) – dictionary of students to search, default: None When `students` is unspecified, all students in gradebook are retrieved.

Raises

- **requests.RequestException** – Exception connection error
- **ValueError** – Unable to decode response content

Returns tuple of student id and student dictionary.

Return type tuple

get_students(*gradebook_id=''*, *simple=False*, *section_name=''*, *include_photo=False*, *include_grade_info=False*, *include_grade_history=False*, *include_makeup_grades=False*)
Get students for a gradebook.

Get a list of students for a given gradebook, specified by a gradebook id. Does not include grade data.

Parameters

- **gradebook_id** (*str*) – unique identifier for gradebook, i.e. `2314`
- **simple** (*bool*) – if `True`, just return dictionary with keys `email`, `name`, `section`, default = `False`
- **section_name** (*str*) – section name
- **include_photo** (*bool*) – include student photo, default= `False`
- **include_grade_info** (*bool*) – include student's grade info, default= `False`
- **include_grade_history** (*bool*) – include student's grade history, default= `False`
- **include_makeup_grades** (*bool*) – include student's makeup grades, default= `False`

Raises

- **requests.RequestException** – Exception connection error
- **ValueError** – Unable to decode response content

Returns list of student dictionaries

Return type list

```
[{
    u'accountEmail': u'stellar.test2@gmail.com',
    u'displayName': u'Molly Parker',
    u'photoUrl': None,
    u'middleName': None,
    u'section': u'Unassigned',
    u'sectionId': 1293925,
    u'editable': False,
    u'overallGradeInformation': None,
```

(continues on next page)

(continued from previous page)

```
    u'studentId': 1145,
    u'studentAssignmentInfo': None,
    u'sortableName': u'Parker, Molly',
    u'surname': u'Parker',
    u'givenName': u'Molly',
    u'nickName': u'Molly',
    u'email': u'stellar.test2@gmail.com'
},]
```

multi_grade(*grade_array*, *gradebook_id=''*)

Set multiple grades for students.

Set multiple student grades for a gradebook. The grades are passed as a list of dictionaries.

Each grade dictionary in `grade_array` must contain a `studentId` and a `assignmentId`. Options for grade mode are: OVERALL_GRADE = `1`, REGULAR_GRADE = `2` To set 'excused' as the grade, enter `None` for `letterGradeValue` and `numericGradeValue`, and pass `x` as the `specialGradeValue`. The `ReturnAffectedValues` flag determines whether to return student cumulative points and impacted assignment category grades (average and student grade)

```
[
    {
        u'comment': None,
        u'booleanGradeValue': None,
        u'studentId': 1135,
        u'assignmentId': 4522,
        u'specialGradeValue': None,
        u'returnAffectedValues': True,
        u'letterGradeValue': None,
        u'mode': 2,
        u'numericGradeValue': 50,
        u'isGradeApproved': False
    },
    {
        u'comment': None,
        u'booleanGradeValue': None,
        u'studentId': 1135,
        u'assignmentId': 4522,
        u'specialGradeValue': u'x',
        u'returnAffectedValues': True,
        u'letterGradeValue': None,
        u'mode': 2,
        u'numericGradeValue': None,
        u'isGradeApproved': False
    },
    {
        u'comment': None,
        u'booleanGradeValue': None,
        u'studentId': 1135,
        u'assignmentId': None,
        u'specialGradeValue': None,
        u'returnAffectedValues': True,
        u'letterGradeValue': u'A',
        u'mode': 1,
        u'numericGradeValue': None,
        u'isGradeApproved': False
    }
```

(continues on next page)

(continued from previous page)

```
]
```

Parameters

- **grade_array** (*dict*) – an array of grades to save
- **gradebook_id** (*str*) – unique identifier for gradebook, i.e. `2314`

Raises

- **requests.RequestException** – Exception connection error
- **ValueError** – Unable to decode response content

Returns dictionary containing response `status` and `message`

Return type dict

set_grade(*assignment_id*, *student_id*, *grade_value*, *gradebook_id=''*, ***kwargs*)

Set numerical grade for student and assignment.

Set a numerical grade for for a student and assignment. Additional options for grade `mode` are: OVERALL_GRADE = `1`, REGULAR_GRADE = `2` To set 'excused' as the grade, enter `None` for letter and numeric grade values, and pass `x` as the `specialGradeValue`. `ReturnAffectedValues` flag determines whether or not to return student cumulative points and impacted assignment category grades (average and student grade).

Parameters

- **assignment_id** (*str*) – numerical ID for assignment
- **student_id** (*str*) – numerical ID for student
- **grade_value** (*str*) – numerical grade value
- **gradebook_id** (*str*) – unique identifier for gradebook, i.e. `2314`
- **kwargs** (*dict*) – dictionary of additional parameters

```
{
    u'letterGradeValue':None,
    u'booleanGradeValue':None,
    u'specialGradeValue':None,
    u'mode':2,
    u'isGradeApproved':False,
    u'comment':None,
    u'returnAffectedValues': True,
}
```

Raises

- **requests.RequestException** – Exception connection error
- **ValueError** – Unable to decode response content

Returns

dictionary containing response `status` and `message`

```
{
    u'message': u'grade saved successfully',
    u'status': 1
}
```

Return type dict

spreadsheet2gradebook(*csv_file*, *email_field=None*, *approve_grades=False*, *use_max_points_column=False*, *max_points_column=None*, *normalize_column=None*)

Upload grade spreadsheet to gradebook.

Upload grades from CSV format spreadsheet file into the Learning Modules gradebook. The spreadsheet must have a column named `External email` which is used as the student's email address (for looking up and matching studentId).

These columns are disregarded: `ID`, `Username`, `Full Name`, `edX email`, `External email`, as well as the strings passed in `max_points_column` and `normalize_column`, if any. All other columns are taken as assignments.

If `email_field` is specified, then that field name is taken as the student's email.

```
External email,AB Assignment 01,AB Assignment 02
jeannechiang@gmail.com,1.0,0.9
stellar.test2@gmail.com,0.2,0.4
stellar.test1@gmail.com,0.93,0.77
```

Parameters

- **csv_reader** (*str*) – filename of csv data, or readable file object
- **email_field** (*str*) – student's email
- **approve_grades** (*bool*) – Should grades be auto approved?
- **use_max_points_column** (*bool*) – If `True`, read the max points and normalize values from the CSV and use the max points value in place of the default if normalized is `False`.
- **max_points_column** (*str*) – The name of the max_pts column. All rows contain the same number, the max points for the assignment.
- **normalize_column** (*str*) – The name of the normalize column which indicates whether to use the max points value.

Raises

- **PyLmodFailedAssignmentCreation** – Failed to create assignment
- **requests.RequestException** – Exception connection error
- **ValueError** – Unable to decode response content

Returns tuple of dictionary containing response `status` and `message`, and duration of operation

Return type tuple

static unravel_sections(*section_data*)

Unravels section type dictionary into flat list of sections with section type set as an attribute.

Parameters **section_data** (*dict*) – Data return from py:method::get_sections

Returns

Flat list of sections with `sectionType` set to type (i.e. recitation, lecture, etc)

Return type list

static unravel_staff(*staff_data*)

Unravels staff role dictionary into flat list of staff members with `role` set as an attribute.

Parameters **staff_data** (*dict*) – Data return from py:method::get_staff

Returns

Flat list of staff members with `role` set to role type (i.e. course_admin, instructor, TA, etc)

Return type list

3.1.3 Membership Class

Contains Membership class

class pylmod.membership.**Membership**(*cert*, *urlbase='https://learning-modules.mit.edu:8443/'*, *uuid=None*)

Bases: *pylmod.base.Base*

Provide API for functions that return group membership data from MIT Learning Modules Web service.

API reference at https://learning-modules-dev.mit.edu/service/membership/doc.html

email_has_role(*email*, *role_name*, *uuid=None*)

Determine if an email is associated with a role.

Parameters

- **email** (*str*) – user email
- **role_name** (*str*) – user role
- **uuid** (*str*) – optional uuid. defaults to self.cuuid

Raises

- **PyLmodUnexpectedData** – Unexpected data was returned.
- **requests.RequestException** – Exception connection error

Returns True or False if email has role_name

Return type bool

get_course_guide_staff(*course_id=''*)

Get the staff roster for a course.

Get a list of staff members for a given course, specified by a course id.

Parameters **course_id** (*int*) – unique identifier for course, i.e. `2314`

Raises

- **requests.RequestException** – Exception connection error
- **ValueError** – Unable to decode response content

Returns

list of dictionaries containing staff data

An example return value is:

```
[
    {
        u'displayName': u'Huey Duck',
        u'role': u'TA',
        u'sortableDisplayName': u'Duck, Huey'
    },
    {
        u'displayName': u'Louie Duck',
        u'role': u'CourseAdmin',
        u'sortableDisplayName': u'Duck, Louie'
    },
    {
        u'displayName': u'Benjamin Franklin',
        u'role': u'CourseAdmin',
        u'sortableDisplayName': u'Franklin, Benjamin'
    },
    {
        u'displayName': u'George Washington',
        u'role': u'Instructor',
        u'sortableDisplayName': u'Washington, George'
    },
]
```

Return type list

get_course_id(*course_uuid*)
Get course id based on uuid.

Parameters **uuid** (*str*) – course uuid, i.e. /project/mitxdemosite

Raises

- **PyLmodUnexpectedData** – No course data was returned.
- **requests.RequestException** – Exception connection error

Returns numeric course id

Return type int

get_group(*uuid=None*)
Get group data based on uuid.

Parameters **uuid** (*str*) – optional uuid. defaults to self.cuuid

Raises

- **PyLmodUnexpectedData** – No data was returned.
- **requests.RequestException** – Exception connection error

Returns group json

Return type dict

get_group_id(*uuid=None*)
Get group id based on uuid.

Parameters **uuid** (*str*) – optional uuid. defaults to self.cuuid

Raises

- **PyLmodUnexpectedData** – No group data was returned.
- **requests.RequestException** – Exception connection error

Returns numeric group id

Return type int

get_membership(*uuid=None*)

Get membership data based on uuid.

Parameters **uuid** (*str*) – optional uuid. defaults to self.cuuid

Raises

- **PyLmodUnexpectedData** – No data was returned.
- **requests.RequestException** – Exception connection error

Returns membership json

Return type dict

CHAPTER 4

Indices and Search

- genindex
- modindex
- search

Python Module Index

p

pylmod.base, 7
pylmod.gradebook, 9
pylmod.membership, 21

Index

B

Base (class in pylmod.base), 7

C

cert (pylmod.base.Base attribute), 7
create_assignment() (pylmod.gradebook.GradeBook method), 9

D

delete() (pylmod.base.Base method), 8
delete_assignment() (pylmod.gradebook.GradeBook method), 10

E

email_has_role() (pylmod.membership.Membership method), 21

G

get() (pylmod.base.Base method), 8
get_assignment_by_name() (pylmod.gradebook.GradeBook method), 11
get_assignments() (pylmod.gradebook.GradeBook method), 11
get_course_guide_staff() (pylmod.membership.Membership method), 21
get_course_id() (pylmod.membership.Membership method), 22
get_gradebook_id() (pylmod.gradebook.GradeBook method), 13
get_group() (pylmod.membership.Membership method), 22
get_group_id() (pylmod.membership.Membership method), 22
get_membership() (pylmod.membership.Membership method), 23
get_options() (pylmod.gradebook.GradeBook method), 13
get_section_by_name() (pylmod.gradebook.GradeBook method), 14
get_sections() (pylmod.gradebook.GradeBook method), 14
get_staff() (pylmod.gradebook.GradeBook method), 15
get_student_by_email() (pylmod.gradebook.GradeBook method), 16
get_students() (pylmod.gradebook.GradeBook method), 17
GradeBook (class in pylmod.gradebook), 9
gradebookid (pylmod.base.Base attribute), 8

M

Membership (class in pylmod.membership), 21
multi_grade() (pylmod.gradebook.GradeBook method), 18

P

post() (pylmod.base.Base method), 8
pylmod.base (module), 7
pylmod.gradebook (module), 9
pylmod.membership (module), 21

R

rest_action() (pylmod.base.Base method), 8
RETRIES (pylmod.base.Base attribute), 7

S

set_grade() (pylmod.gradebook.GradeBook method), 19
spreadsheet2gradebook() (pylmod.gradebook.GradeBook method), 20

T

TIMEOUT (pylmod.base.Base attribute), 8

U

unravel_sections() (pylmod.gradebook.GradeBook static method), 20

unravel_staff() (pylmod.gradebook.GradeBook static method), 21
urlbase (pylmod.base.Base attribute), 7

V

verbose (pylmod.base.Base attribute), 9