lectio.py

Release 0.2.1

dnorhoj

Sep 29, 2022

CONTENTS

Lectio.py is a Python library for reading and interacting with lectio.dk, a Danish school management system.

Note: This library is not affiliated with lectio.dk or macom in any way. Nor is it endorsed by them. It scrapes the website as there is no official API.

Features:

- Pythonic API
- Easy to use

CHAPTER

ONE

INFORMATION

Useful links: *Installation* | *Quickstart*

Reference: *API Reference* | *Exceptions*

CHAPTER

TWO

TABLE OF CONTENTS

2.1 Installation

To install lectio.py and its dependencies, you can simply install it from PyPi:

```
pip install lectio.py
```

or clone the repository and install it manually:

```
git clone https://github.com/dnorhoj/Lectio.py.git
cd lectio.py
python setup.py install
```

You can check if the installation was successful by running:

```
python

>>> import lectio
>>> lectio.__version__
```

2.2 Quickstart

Let's start with a simple example, where we will get your schedule for today.

2.2.1 Schedule

Create a new file, and add the following code:

```
from lectio import Lectio
from datetime import datetime, timedelta

lec = Lectio('<username>', '<password>')

# Get my user object
me = lec.me()

# Get the schedule for today
schedule = me.get_schedule(datetime.now(), datetime.now() + timedelta(days=1))
```

We now have a list of all *lectio.helpers.schedule.Module* in the `schedule` variable. Let's print it:

```
for module in schedule:
    print(module.title, module.teacher, module.room)
```

More to come...!

2.3 API Reference

class lectio.**Lectio**(*inst_id: int*)

The main Lectio class.

A Lectio object is your gateway to manipulating and getting data from Lectio.

Parameters

inst_id (*int*) – Your Lectio institution id.

You can find this by going to your institution's Lectio login page and it should be in the URL like this:

```
https://www.lectio.dk/lectio/123/login.aspx
```

Here, the `123` would be my institution id.

authenticate(*username: str*, *password: str*, *save_creds: bool = True*) → bool

Authenticates you on Lectio.

Note: Running *authenticate()* on an already authenticated object will log you out of the already authenticated user.

This will happen even though authentication was unsuccessful.

Parameters

- **username** (*str*) – Lectio username for the given institution id.
- **password** (*str*) – Lectio password for the given institution id.
- **save_creds** (*bool*) – Whether the credentials should be saved in the object (useful for auto relogin on logout)

Raises

- *exceptions.IncorrectCredentialsError* – When incorrect credentials passed
- *exceptions.InstitutionDoesNotExistError* – When the institution id passed on creation of object is invalid

Example:

```
from lectio import Lectio, exceptions

lect = Lectio(123)

try:
```

(continues on next page)

(continued from previous page)

```
    lect.authenticate("username", "password")
    print("Authenticated")
except exceptions.IncorrectCredentialsError:
    print("Not authenticated")
    exit(1)

...
```

log_out() → None

Clears entire session, thereby logging you out

Returns
None

me() → *Me*

Gets the authenticated user

Returns
Own user object

Return type
lectio.models.user.Me

school() → *School*

Returns a *lectio.models.school.School* object for the given institution id.

Returns
The school object for the authenticated user.

Return type
lectio.models.school.School

class lectio.models.school.**School**(*lectio:* Lectio)

A school object.

Represents a school.

Note: This class should not be instantiated directly, but rather through the `lectio.Lectio.get_school()` method.

Parameters
lectio (*lectio.Lectio*) – Lectio object

get_all_students() → List[*User*]

Get all students

Returns
List of students

Return type
list(`User`)

get_students_by_letter(*letter: str*) → List[*User*]

Get students by first letter of name

Parameters
letter (*str*) – Letter to search for

Returns
List of students

Return type
list(*lectio.models.user.User*)

get_teachers() → List[*User*]

Get all teachers

Returns
List of teachers

Return type
list(*lectio.models.user.User*)

get_user_by_id(*user_id: str*, *user_type:* UserType = *UserType.STUDENT*, *check: bool = True*) → *User*

Gets a user by their id

Parameters

- **user_id** (*str*) – The id of the user
- **user_type** (*lectio.models.user.UserType*) – The type of the user (student or teacher)
- **check** (*bool*) – Whether to check if the user exists (slower)

Returns
User object

Return type
lectio.models.user.User

Raises
lectio.exceptions.UserDoesNotExistError – When the user does not exist

search_for_students(*query: str*) → List[*User*]

Search for user

Note: This method is not very reliable, and will sometimes return no results. Also, the query has to be from the beginning of the name.

Example: Searching for "John" will return "John Doe", but searching for "Doe" might not.

Parameters
query (*str*) – Name to search for

Returns
List of users

Return type
list(lectio.User)

search_for_teachers(*query_name: str*, *query_initials: Optional[str] = None*) → List[*User*]

Search for teachers by name or initials

Parameters

- **query_name** (*str*) – Name to search for
- **query_initials** (*Optional[str]*) – Initials to search for

Returns
List of teachers

Return type
list(*lectio.models.user.User*)

search_for_users(*query: str*) → List[*User*]

Search for user

Parameters
query (*str*) – Name to search for

Returns
List of users

Return type
list(*lectio.models.user.User*)

2.3.1 User

class lectio.models.user.**User**(*lectio:* Lectio, *user_id: int*, *user_type:* UserType = *UserType.STUDENT*, ***, *lazy=False*, ***user_data*)

Lectio user object

Represents a lectio user

Note: This class should not be instantiated directly, but rather through the `lectio.Lectio.get_user()` or *lectio.models.school.School.search_for_users()* methods or similar.

Parameters

- **lectio** (*lectio.Lectio*) – Lectio object
- **user_id** (*int*) – User id
- **user_type** (*lectio.models.user.UserType*) – User type (UserType.STUDENT or UserType.TEACHER)
- **lazy** (*bool*) – Whether to not populate user object on instantiation (default: False)

id

User id

Type
int

type

User type (UserType.STUDENT or UserType.TEACHER)

Type
lectio.models.user.UserType

property class_name: str

User's class name (only for students)

Type
str|None

get_schedule(*start_date: datetime*, *end_date: datetime*, *strip_time: bool = True*) → List[*Module*]

Get schedule for user

Note: As lectio is weird, you can only get a schedule for a range that is less than one month. If you specify a range greater than one month, you will get an empty return list.

Parameters

- **start_date** (datetime.datetime) – Start date
- **end_date** (datetime.datetime) – End date
- **strip_time** (*bool*) – Whether to remove hours, minutes and seconds from date info, also adds 1 day to end time. Basically just allows you to put in a random time of two days, and still get all modules from all the days including start and end date.

property image: str

User's image url

Type
str

property initials: str

User's initials (only for teachers)

Type
str|None

property name: str

User's name

Type
str

class lectio.models.user.**Me**(*lectio: Lectio*, *user_id: int*, *user_type: UserType = UserType.STUDENT*, ***, *lazy=False*, ***user_data*)

class lectio.models.user.**UserType**(*value*)

User types enum

Example

```
>>> from lectio import Lectio
>>> from lectio.models.user import UserType
>>> lec = Lectio(123)
>>> lec.authenticate("username", "password")
>>> me = lec.me()
>>> print(me.type)
0
```

(continues on next page)

(continued from previous page)

```
>>> print(me.type == UserType.STUDENT)
True
```

STUDENT = 0

TEACHER = 1

get_str() → str

Get string representation of user type for lectio interface in english

Returns

String representation of user type

Return type

str

2.3.2 Misc

class lectio.helpers.schedule.**Module**(***kwargs*)

Lectio module object

Represents a lectio module

Parameters

- **title** (*str* | *None*) – Optional description of module (not present in all modules)
- **subject** (*str* | *None*) – "Hold" from lectio, bascially which subject. Example: *1.a Da*
- **teacher** (*str* | *None*) – Initials of teacher. Example: *abcd*
- **room** (*str* | *None*) – Room name of module. Example: *0.015*
- **extra_info** (*str* | *None*) – Extra info from module, includes homework and other info.
- **start_time** (datetime.datetime) – Start time of module
- **end_time** (datetime.datetime) – End time of module
- **status** (*int*) – 0=normal, 1=changed, 2=cancelled
- **url** (*str* | *None*) – Url for more info for the module

display()

2.4 Exceptions

Here are all the custom Lectio.py exceptions as well as their explanation

exception lectio.exceptions.**IncorrectCredentialsError**

Incorrect credentials error, mostly thrown in auto-login on session expired

exception lectio.exceptions.**InstitutionDoesNotExistError**

The institution with the id you provided does not exist.

exception lectio.exceptions.**LectioError**

Base lectio.py exception

exception lectio.exceptions.**UnauthenticatedError**

Throws when trying to get data while unauthenticated or session expired

exception lectio.exceptions.**UserDoesNotExistError**

The user does not exist.

PYTHON MODULE INDEX

INDEX