Penn SDK Documentation

Release 1.6.8

Penn Labs

Feb 10, 2018

Contents

Release v1.6.8.

Penn SDK is the Python library for writing code that interfaces with University of Pennsylvania data. It consists of wrappers for the Registrar, Dining, and Directory API's

Contents:

CHAPTER 1

Registrar API Wrapper

This documentation explains the methods of the *Registrar* object. For the specific fields on the objects returned by these methods, see the official documentation.

Note: The Penn Data Warehouse performs nightly maintenance, which causes downtime for Penn InTouch and the Penn OpenData Registrar API. This will throw an `APIError` in the SDK, which client applications can catch and try to fetch the data again later.

class penn.registrar.**Registrar**(*bearer*, *token*)

The client for the Registrar. Used to make requests to the API.

Parameters

- **bearer** – The user code for the API
- **token** – The password code for the API

Usage:

```
>>> from penn import Registrar
>>> r = Registrar('MY_USERNAME_TOKEN', 'MY_PASSWORD_TOKEN')
```

course(*dept*, *course_number*)

Return an object of semester-independent course info. All arguments should be strings.

```
>>> cis120 = r.course('cis', '120')
```

department(*dept*)

Return an iterator of all course-info objects in a department, in no particular order.

search(*params*, *validate=False*)

Return a generator of section objects for the given search params.

Parameters

- **params** – Dictionary of course search parameters.

- **validate** – Optional. Set to true to enable request validation.

```
>>> cis100s = r.search({'course_id': 'cis', 'course_level_at_or_below': '200'}
↪)
```

search_params()
Return a dictionary of possible search parameters and their possible values and descriptions.

section(*dept*, *course_number*, *sect_number*)
Return a single section object for the given section. All arguments should be strings. Throws a *ValueError* if the section is not found.

```
>>> lgst101_bfs = r.course('lgst', '101', '301')
```

CHAPTER 2

Dining API Wrapper

This documentation explains the methods of the *Dining* object. For the specific fields on the objects returned by these methods, see the official documentation.

class penn.dining.**Dining**(*bearer*, *token*)

The client for the Registrar. Used to make requests to the API.

Parameters

- **bearer** – The user code for the API
- **token** – The password code for the API

Usage:

```
>>> from penn import Dining
>>> din = Dining('MY_USERNAME_TOKEN', 'MY_PASSWORD_TOKEN')
```

menu_daily(*building_id*)

Get a menu object corresponding to the daily menu for the venue with building_id.

Parameters **building_id** – A string representing the id of a building, e.g. "abc".

```
>>> commons_today = din.menu_daily("593")
```

menu_weekly(*building_id*)

Get an array of menu objects corresponding to the weekly menu for the venue with building_id.

Parameters **building_id** – A string representing the id of a building, e.g. "abc".

```
>>> commons_week = din.menu_weekly("593")
```

venues()

Get a list of all venue objects.

```
>>> venues = din.venues()
```

CHAPTER 3

Directory API Wrapper

This documentation explains the methods of the *Directory* object. For the specific fields on the objects returned by these methods, see the official documentation.

class penn.directory.**Directory**(*bearer*, *token*)

The client for the Directory. Used to make requests to the API.

Parameters

- **bearer** – The user code for the API
- **token** – The password code for the API

Usage:

```
>>> from penn import Directory
>>> d = Directory('MY_USERNAME_TOKEN', 'MY_PASSWORD_TOKEN')
```

detail_search(*params*, *standardize=False*)

Get a detailed list of person objects for the given search params.

Parameters **params** – Dictionary specifying the query parameters

```
>>> people_detailed = d.detail_search({'first_name': 'tobias', 'last_name':
↪'funke'})
```

person_details(*person_id*, *standardize=False*)

Get a detailed person object

Parameters **person_id** – String corresponding to the person's id.

```
>>> instructor = d.person('jhs878sfd03b38b0d463b16320b5e438')
```

search(*params*, *standardize=False*)

Get a list of person objects for the given search params.

Parameters

- **params** – Dictionary specifying the query parameters

- **standardize** – Whether to standardize names and other features, currently disabled for backwards compatibility. Currently standardizes names, lowercases emails, and removes faculty label from affiliation.

```
>>> people = d.search({'first_name': 'tobias', 'last_name': 'funke'})
```

CHAPTER 4

Transit API Wrapper

This documentation explains the methods of the *Transit* object. For the specific fields on the objects returned by these methods, see the official documentation.

class penn.transit.**Transit**(*bearer*, *token*)

The client for Transit. Used to make requests to the API.

Parameters

- **bearer** – The user code for the API
- **token** – The password code for the API

Usage:

```
>>> from penn import Transit
>>> trans = Transit('MY_USERNAME_TOKEN', 'MY_PASSWORD_TOKEN')
```

apc(*start_date*, *end_date*)

Return all APC data packets in date range

Parameters

- **start_date** – The starting date for the query.
- **end_date** – The end date for the query.

```
>>> import datetime
>>> today = datetime.date.today()
>>> trans.apc(today - datetime.timedelta(days=1), today))
```

configuration()

Return route configuration info

```
>>> route_config = trans.configuration()
```

mdt(*start_date*, *end_date*)

Return all MDT data packets in date range

Parameters

- **start_date** – The starting date for the query.
- **end_date** – The end date for the query.

```
>>> import datetime
>>> today = datetime.date.today()
>>> trans.mdt(today - datetime.timedelta(days=1), today))
```

prediction()
Return route data and time predictions

```
>>> predictions = trans.prediction()
```

stopinventory()
Return a list all transit stops.

```
>>> stops = trans.stopinventory()
```

stoptimes(*start_date*, *end_date*)
Return all stop times in the date range

Parameters

- **start_date** – The starting date for the query.
- **end_date** – The end date for the query.

```
>>> import datetime
>>> today = datetime.date.today()
>>> trans.stoptimes(today - datetime.timedelta(days=1), today)
```

transapc(*start_date*, *end_date*)
Return detail of boardings, alightings, by vehicle and stop, including the passenger load leaving the stop (this is only for vehicles equipped with APC hardware)

Parameters

- **start_date** – The starting date for the query.
- **end_date** – The end date for the query.

```
>>> import datetime
>>> today = datetime.date.today()
>>> trans.transapc(today - datetime.timedelta(days=1), today))
```

CHAPTER 5

Map API Wrapper

This documentation explains the methods of the *Map* object. For the specific fields on the objects returned by these methods, see the official documentation.

class penn.map.**Map**(*bearer*, *token*)

The client for the Map Search API.

Parameters

- **bearer** – The user code for the API
- **token** – The password code for the API

Usage:

```
>>> from penn import Map
>>> n = Map('MY_USERNAME_TOKEN', 'MY_PASSWORD_TOKEN')
```

search(*keyword*)

Return all buildings related to the provided query.

Parameters **keyword** – The keyword for your map search

```
>>> results = n.search('Harrison')
```

CHAPTER 6

News API Wrapper

This documentation explains the methods of the *News* object. For the specific fields on the objects returned by these methods, see the official documentation.

class penn.news.**News**(*bearer*, *token*)

The client for the News Search API.

Parameters

- **bearer** – The user code for the API
- **token** – The password code for the API

Usage:

```
>>> from penn import News
>>> n = News('MY_USERNAME_TOKEN', 'MY_PASSWORD_TOKEN')
```

search(*keyword*)

Return all news related to the provided query.

Parameters **keyword** – The keyword for your news search

```
>>> results = n.search('interview')
```

CHAPTER 7

Laundry Website Scraper

This documentation explains the methods of the *Laundry* object. This SDK module is implemented by scraping the laundry alert service.

class penn.laundry.**Laundry**

The client for Laundry. Used to make requests to the API.

Usage:

```
>>> from penn import Laundry
>>> l = Laundry()
```

all_status()

Return names, hall numbers, and the washers/dryers available for all rooms in the system

```
>>> all_laundry = l.all_status()
```

create_hall_to_link_mapping()

Returns Mapping from hall name to associated link in SUDS. Creates inverted index from id to hall.

hall_status(*hall_id*)

Return the status of each specific washer/dryer in a particular laundry room.

Parameters **hall_id** – Integer corresponding to the id of the hall. This id is returned as part of the all_status call.

```
>>> english_house = l.hall_status("English%20House")
```

machine_usage(*hall_no*)

Returns the average usage of laundry machines every hour for a given hall.

The usages are returned in a dictionary, with the key being the day of the week, and the value being an array listing the usages per hour.

Parameters **hall_no** – integer corresponding to the id number for the hall. Thus number is returned as part of the all_status call.

```
>>> english_house = l.machine_usage(2)
```

parse_a_hall (*hall*)

Return names, hall numbers, and the washers/dryers available for a certain hall.

Parameters **hall** (*int*) – The ID of the hall to retrieve data for.

CHAPTER 8

Library Website Scraper

This documentation explains the methods of the *StudySpaces* object. This SDK module is implemented by scraping the Penn Libraries LibCal website.

class penn.studyspaces.**StudySpaces**

Used for interacting with the UPenn library GSR booking system.

Usage:

```
>>> from penn import StudySpaces
>>> s = StudySpaces()
```

book_room(*building*, *room*, *start*, *end*, *firstname*, *lastname*, *email*, *groupname*, *phone*, *size*, *fake=False*)

Books a room given the required information.

Parameters

- **building** (*int*) – The ID of the building the room is in.
- **room** (*int*) – The ID of the room to book.
- **start** (*datetime*) – The start time range of when to book the room.
- **end** (*datetime*) – The end time range of when to book the room.
- **fake** (*bool*) – If this is set to true, don't actually book the room. Default is false.

Returns Boolean indicating whether the booking succeeded or not.

Raises **ValueError** – If one of the fields is missing or incorrectly formatted, or if the server fails to book the room.

get_buildings()

Returns a list of building IDs, building names, and services.

static get_room_id_name_mapping(*building*)

Returns a dictionary mapping id to name, thumbnail, and capacity.

The dictionary also contains information about the lid and gid, which are used in the booking process.

Parameters **building** (*int*) – The ID of the building to fetch rooms for.

Returns A list of rooms, with each item being a dictionary that contains the room id and available times.

get_rooms(*building*, *start*, *end*)
Returns a dictionary matching all rooms given a building id and a date range.

The resulting dictionary contains both rooms that are available and rooms that already have been booked.

Parameters

- **building** (*int*) – The ID of the building to fetch rooms for.
- **start** (*datetime*) – The start date of the range used to filter available rooms.
- **end** (*datetime*) – The end date of the range used to filter available rooms.

static parse_date(*date*)
Converts library system dates into timezone aware Python datetime objects.

Parameters **date** (*datetime*) – A library system date in the format '2018-01-25 12:30:00'.

Returns A timezone aware python datetime object.

CHAPTER 9

Penn Academic Calendar API

This documentation explains the methods of the *Calendar* object.

class penn.calendar3year.**Calendar**

pull_3year()

Returns a list (in JSON format) containing all the events from the Penn iCal Calendar.

List contains events in chronological order.

Each element of the list is a dictionary, containing:

- Name of the event 'name'
- Start date 'start'
- End date 'end'

Python Module Index

p

Index

V