LabStats API

Mar 05, 2020

Contents:

LabStats API is an unofficial Python wrapper for the LabStats REST API. It can be used to quickly and easily analyze data from LabStats.

CHAPTER 1

Install

LabStats API requires Python 3.6 or higher.

You can install LabStats API. . .

Via pipenv:

```
pipenv install git+https://bitbucket.org/cvtc-desktopservices/labstats_api.git
↪#egg=labstats_api
```

Via pip:

```
pip install git+https://bitbucket.org/cvtc-desktopservices/labstats_api.git
↪#egg=labstats_api
```

CHAPTER 2

Usage

LabStats API is designed to be a simple way to get to the data you need in LabStats. It takes away all of the hard work of interacting with the API, handling authentication, and working with pages.

2.1 Getting started

To use LabStats API, we'll start by instantiating a LabStats object:

```
from os import environ
from labstats_api import LabStats

labstats = LabStats(api_url, api_key)
```

`api_url` is your LabStats URL, such as `https://api.labstats.com`, and `api_key` is your API token. You can create one in "Admin -> External Systems" in LabStats.

Now that we have the LabStats object, we can make calls as specified on the page *The LabStats object*. For example:

```
apps = labstats.get_apps()

for app in apps:
    print(app.name)

>>> 'Microsoft Word'
>>> 'Microsoft Excel'
```

You can pass arbitrary parameters to any method as keyword arguments and they will be added to your query. For more information, see *Keyword arguments*.

2.2 Working with LabStats Objects

Whenever you create a query with LabStats API, you will receive either a single object which is a subclass of `LabStatsObject` or an `AfterIDPaginatedList` which contains `LabStatsObject`'s. This makes accessing data easy.

2.2.1 LabStatsObject

LabStats API converts JSON objects from the LabStats REST API into Python objects. This means that you can access the data of an object without hassling with `json.loads()` or `object["attribute"]`.

For example, `Applications` in the REST API have a `vendor` attribute. So, once we have an `Application`, we can access this attribute:

```
app.vendor
>>> Microsoft Corporation
```

We've attempted to document all of these attributes in *API Reference*, but the REST API may change without notice. In this case, you can simply use the new attribute with no changes to LabStats API. For example, if a new `color` attribute was added to `Applications`, `Application.color` would instantly be available to your code.

2.2.2 AfterIDPaginatedList

The `labstats.paginated_list.AfterIDPaginatedList` is an abstraction of the LabStats REST API's approach to serving multiple pages of results. When it runs out of locally loaded results, it will load more with another request to the API. For most purposes, you can treat it as a regular list:

```
apps = labstats.get_apps()
apps[0].name

>>> 'Microsoft Word'

apps[1].name

>>> 'Microsoft Excel'
```

The list may also be sliced:

```
slice = apps[0:10]
for app in slice:
    print(app.name)

>>> Microsoft Word
>>> Microsoft Excel
>>> Microsoft PowerPoint
>>> Microsoft Outlook
>>> Microsoft Publisher
>>> Microsoft Internet Explorer
>>> Mozilla Firefox
>>> Google Chrome
>>> Notepad++
>>> Notepad
```

2.3 Keyword arguments

Almost all calls to LabStats API create a query to the LabStats REST API. These queries can take a number of parameters which are specified in the LabStats REST API documentation. In every case, we have attempted to document these parameters in the *API Reference*, but the REST API is subject to change with no (or little) notice. If you see a parameter that you want to add to your query but you don't see it in the API reference, you can add it as an arbitrary keyword argument.

For example, suppose that LabStats added a `color` parameter to the `/apps` endpoint. You notice this on the LabStats REST API Documentation and would like to filter your searches to applications which are blue. To do this:

```
labstats.get_applications(color="blue")
```

2.4 "Gotchas"

These are interesting or potentially surprising things you may encounter while using LabStats API.

2.4.1 Limit

Many API calls have a `limit` parameter which sets the number of items to request per page returned. We've tried to stick with the default values set by LabStats when dealing with these values, but you may want to tune them.

If you will only ever use a few items of a query, it does not make sense to ask your LabStats instance for more. You'll slightly reduce your bandwidth and memory requirements by setting a lower value for the limit and staying under that value when using the resulting list. If you go over your limit in the resulting list, you'll end up making a new request to the REST API which will take a small amount of time.

For example, setting a limit of 25 and using items 0-24 of the resulting list is optimal. A request for item 25 will cause the list to load more items. Requesting a very high item, such as item 300, will cause the list to load multiple pages in an attempt to get to your requested item. In this case it is better to set a higher limit (or try to filter using other parameters in the query so you don't need to request so many items).

2.4.2 Results do not update automatically

If you request an item using LabStats API then change the item's data in LabStats, this change will not be reflected in your local object. You will need to run your original query again to get updated data. Luckily, most calls in LabStats API accept either an integer ID or the same objects that they return as parameters to get new data. For example:

```
app = labstats.get_app(1000)
app.name

>>> Microsoft Word

# Change a few things using the LabStats UI...
app.name

>>> Microsoft Word

app = labstats.get_app(app)
app.name

>>> Neat Word Processing
```

Similarly, lists of results may be changed on the LabStats instance between the time it is first requested and when you ask for more data. To reduce the risk of this causing issues for you, set your *Limit* to an appropriate value and request as many results as you need quickly after requesting the list. For example, if you will use an entire list of applications, request and then fill the list immediately:

```
apps = labstats.get_applications()
[__ for __ in apps]
```

This list comprehension iterates through every value in the list and does nothing with it, filling the list. Whenever the list is used after this point, each result is pulled from memory rather than from the REST API.

2.4.3 The After ID

Calls which return an *AfterIDPaginatedList* take an `after_id` argument. This argument is used in the REST API's pagination, but you can also use it in limited cases. The `after_id` says "I would like all of the objects which have an ID greater than this one." In other words, if you pass `1001`, you'll get objects with ID `1002` and up. If you don't have an object with ID `1002`, you'll receive the next smallest ID after `1001`.

CHAPTER 3

API Reference

This section contains reference material for LabStats API. Note that the documentation for this Python wrapper may fall behind the real API, so it's best to also have the LabStats REST API Documentation open for reference.

If you'd like a prose introduction to LabStats API, see *Usage*. Otherwise, *The LabStats object* is probably where you want to start.

3.1 The LabStats object

The LabStats API "Starter Session". Start here to get anywhere in the API

exception labstats_api.labstats.**KeyNotGivenError**
Thrown when no API key is given

class labstats_api.labstats.**LabStats**(*api_url*, *api_key*, *session=None*)
A LabStats API session.

The LabStats object is the "starter" object for using LabStats API. Most API operations will start from here.

See the LabStats REST API Documentation for more information on REST API calls. See *Usage* for information on using LabStats API.

Parameters

- **api_url** – URL of your LabStats server's API. If you have a hosted instance, see API Documentation and Testing. Otherwise, your API URL will be specific to your self-hosted instance.
- **api_key** – API authorization key to query the LabStats server with. See API Key Creation for more information.
- **session** – A Requests Session object to use for this instance. See Advanced Usage – Requests for more information. If you're unsure if you need this, you probably don't. If this option is omitted (or is `None`), a Session will be created for you.

get_app(*application*, ***kwargs*)
Gets a single `[Application]` from a LabStats account

Parameters **application** (*labstats_api.models.Application* or int) – The application to query for.

Returns *labstats_api.models.Application*

Calls `/apps/{id}/`

get_app_tags(***kwargs*)
Gets a list of all of the application tags from a LabStats account

Returns list of `str`

Calls `/apps/tags/`

get_apps(*type=None*, *search=None*, *limit=300*, *after_id=1*, ***kwargs*)
Gets a list of all the Applications from a LabStats account

Parameters

- **type** – The type of application to receive. Valid values are `"catalog"`, `"inventory"`, or `"user_defined"`.
- **search** (*str*) – A search term used to filter applications
- **limit** – See *Limit*
- **after_id** – See *The After ID*

Returns list of *labstats_api.models.Application*

Calls `/apps/`

Additional parameters to the request can be passed as `kwargs`

get_group(*group*, ***kwargs*)
Gets a single Group from a LabStats account

Parameters **group** (*models.Group* or int) – Group to query for

Returns *models.Group*

Calls `/groups/{id}/`

get_groups(*search=None*, *contents=None*, *has_lab_features_enabled=None*, *capability=None*, ***kwargs*)
Gets a list of all the Groups from a LabStats account

Parameters

- **search** – Search term
- **contents** – `"stations"` or `"groups"`
- **has_lab_features_enabled** – Whether lab features are enabled or not
- **capability** – Search for a single lab capability

Returns list of *models.Group*

Calls `/groups/`

get_station(*station*, ***kwargs*)
Gets a single Station from a LabStats account

Parameters **group** (*models.Group* or int) – Group to query for

Returns *models.Group*

Calls `/stations/{id}/`

get_station_tag_groups(***kwargs*)
Gets a list of Station Tag groups which organize Tags for Stations.

Return type List of *models.StationTagGroup*

Calls `/stations/tag_groups`

get_station_tags(***kwargs*)
Gets all of the Station Tags which organize Stations

Return type List of str

Calls `/stations/tags`

get_stations(*status=None*, *os=None*, *form_factor=None*, *has_application=None*, *limit=200*, *after_id=1*, ***kwargs*)
Gets a list of all the Stations from a LabStats account.

Parameters

- **status** – Filter based on the state of the machine. Valid values are `"powered_on"`, `"offline"`, `"in_use"`, or `None`.
- **os** – Filter based on the machine's operating system. Valid values are `"windows"`, `"macos"`, or `None`.
- **form_factor** – Filter based on the machine's form factor. Valid values are `"desktop"` or `"laptop"`.
- **has_application** (int or *models.Application*) – Filter based on whether the machine has this application installed.
- **limit** – See *Limit*
- **after_id** – See *The After ID*

get_stations_with_tag(*tag*, *limit=200*, *after_id=1*, ***kwargs*)
Gets all of the stations that have the given tag.

exception labstats_api.labstats.**URLNotGivenError**
Thrown when no API URL is given

3.2 Models

3.2.1 Application

class labstats_api.models.**Application**(*requester*, *attributes*)
Bases: *labstats_api.models.LabStatsObject*

Represents a LabStats application

description = None
User-defined Application description

endpoint
The LabStats REST API endpoint that represents this object

get_versions(*limit=300*, *after_id=1*, ***kwargs*)
Gets a list of detected versions for this application

Parameters

- **limit** – See *Limit*
- **after_id** – See *The After ID*

id = None
Integer unique identifier

is_tracked = None
Whether LabStats is tracking usage of this Application

name = None
Human-readable name of the Application, such as `"Notepad"`

source = None
`"catalog"`, `"inventory"`, or `"user_defined"`. Used as `type` parameter for the `/apps` endpoint.

tracking_pattern_match = None
`"wildcard"` or `"reg_ex"`

type = None
`"desktop"` or `"webapp"`

vendor = None
Human-readable name of the Application's vendor

3.2.2 ApplicationVersion

class labstats_api.models.**ApplicationVersion**(*requester*, *attributes*)
Bases: *labstats_api.models.LabStatsObject*

Represents a single version of an Application tracked by LabStats

application_id = None
The ID of the Application which this Version pertains to

id = None
Integer unique identifier

install_count = None
The number of computers that this version was detected on

name = None
Human-readable name of the application at this version

operating_system = None
The platform that this version was detected on

version = None
The application's version string, such as `1.0.0`

3.2.3 Group

class labstats_api.models.**Group**(*requester*, *attributes*)
Bases: *labstats_api.models.LabStatsObject*

Represents a group of Stations or Groups

contents = None
"stations" or "groups"

description = None
Optional longer description

endpoint
The LabStats REST API endpoint that represents this object

get_child_groups(***kwargs*)
Gets a list of Groups which are children of this Group

Returns list of *Group*

get_stations(*status=None*, *limit=200*, *after_id=1*, ***kwargs*)
Returns all Stations inside this group.

If this group contains groups, returns the status of all stations within the groups within this group. Recursively.

Parameters

- **status** – Filter based on the state of the machine. Valid values are "powered_on", "offline", "in_use", or None.
- **limit** – See *Limit*
- **after_id** – See *The After ID*

Return type list of *Station*

Calls /groups/{id}/stations

get_status(***kwargs*)
Returns the status of all stations inside this group.

If this group contains groups, returns the status of all stations within the groups within this group. Recursively.

Return type *GroupStatus*

has_lab_features_enabled = None
Whether Lab Features are enabled

id = None
Integer unique identifier

name = None
Human-readable group name

parent_group_id = None
Identifier of the group containing this group

schedule_id = None
Identifier of the Schedule assigned to this Group

3.2.4 GroupStatus

class labstats_api.models.**GroupStatus**(*offline*, *powered_on*, *in_use*)
Represents the status of Stations in a Group.

offline
Count of the stations which are have not checked in for a while, for example because they are off or asleep.

powered_on
Count of the stations which are checking in but not in use.

in_use
Count of the stations which are checking in and have an active user logged in.

3.2.5 Station

class labstats_api.models.**Station**(*requester*, *attributes*)
Bases: *labstats_api.models.LabStatsObject*

Represents a computer Station.

allow_student_routing = None
Whether the LabFind app should allow users to navigate to this lab.

client_version = None
Version of LabStats client

description = None
Optional user-defined description

endpoint
The LabStats REST API endpoint that represents this object

form_factor = None
`"Desktop"` or `"Laptop"`

get_apps(*limit=300*, *after_id=1*, ***kwargs*)
Gets all the apps installed on this station.

Parameters

- **limit** – See *Limit*
- **after_id** – See *The After ID*

Returns list of *Application*

Calls `/stations/{id}/apps`

get_metadata(***kwargs*)
Gets a list of *StationMetadataAttribute* for this Station.

Return type list of *StationMetadataAttribute*

Calls `/stations/{id}/metadata`

get_status(***kwargs*)
Gets this station's status, whether it's on and in use.

Return type *StationStatus*

Calls `/stations/{id}/status`

group_id = None
Group which this Station belongs to

host_name = None
Network hostname.

Not necessarily the same as *name*. May be a FQDN or not depending on the client's operating system or configuration.

id = None
Unique identifier

ip_addresses = None
List of IPv4 addresses currently granted

ip_v6_addresses = None
List of IPv6 addresses currently granted

mac_addresses = None
List of MAC addresses used by this station

manufacturer = None
The machine's detected manufacturer

model = None
The machine's detected model

name = None
Human-readable machine name. Not necessarily the hostname!

operating_systems = None
List of operating systems the machine has reported with.

An operating system dict looks like this:

```
{
    "name": "windows",
    "version": "10.0.177653"
}
```

A Station may have one or more operating systems, but the only supported ones are `"windows"` and `"macos"`.

serial_number = None
The machine's detected serial number

subnets = None
List of IPv4 subnet masks for ip_addresses

3.2.6 StationMetadataAttribute

class labstats_api.models.**StationMetadataAttribute**(*name*, *value*, *is_user_defined*)
Represents a single attribute of extra or user-defined metadata on a Station.

name
The name of the attribute, such as `"HardDisk_TotalBytes"`.

value
The value contained by the attribute, such as `"250 GB"`.

is_user_defined
A boolean value for whether this attribute was imported by the user or created by the LabStats client.

3.2.7 StationStatus

class labstats_api.models.**StationStatus**
Represents whether a station is powered on and in use

in_use = 'in_use'
The station is turned on and a user is actively using it

offline = 'offline'
The station is turned off or is otherwise not checking in to LabStats

powered_on = 'powered_on'
The station is turned on and has no active user

3.2.8 StationTagGroup

class labstats_api.models.**StationTagGroup**(*requester*, *name*)
Represents a group of tags which may be applied to a Station

endpoint
The endpoint which may be used to get this tag group

name
Unique name. Also the unique identifier for the tag group.

tags
The tags which this tag group contains

Calls `/stations/tags_groups/{tag_group}/tags`

3.3 API Errors

Generic errors, or errors that should not be module-local

exception labstats_api.api_errors.**LabStatsAPIError**
Bases: `Exception`

Base exception for LabStats API.

Thrown in this regular form when no more specific error applies.

exception labstats_api.api_errors.**NotFoundError**
Bases: *labstats_api.api_errors.LabStatsAPIError*

Thrown by LabStats when the requested item was not found

exception labstats_api.api_errors.**RequestInvalidError**
Bases: *labstats_api.api_errors.LabStatsAPIError*

Thrown by LabStats when the request is invalid

exception labstats_api.api_errors.**UnauthorizedError**
Bases: *labstats_api.api_errors.LabStatsAPIError*

Thrown by LabStats when the given API key is invalid or has insufficient permissions.

May also be thrown if the IP you are accessing the API from is not whitelisted for the key.

3.4 Internal use

These objects are meant only for internal use by LabStats API but their documentation is included here for completeness (or if you want to contribute)

class labstats_api.models.**LabStatsObject**(*requester*, *attributes*)

Base class for all classes representing objects returned by the API. This makes a call to *labstats_api.models.LabStatsObject.set_attributes()* to dynamically construct this object's attributes with a JSON object.

Parameters

- **requester** (*labstats_api.requester.Requester*) – The requester to pass HTTP requests through.
- **attributes** (*dict*) – The JSON object to build this object with.

set_attributes(*attributes*)

Load this object with attributes. This method attempts to detect special types based on the field's content and will create an additional attribute of that type.

Consider a JSON response with the following fields:

```
{
    "id": 1000
    "start_time": "2019-06-20T11:04:36.967"
    "name": "Microsoft Word"
}
```

The `start_time` field matches a date in RFC3339 format, so an additional datetime attribute is created, `start_time_date`. `start_time` will hold the original string representing the date. `start_time_date` contains a datetime object representing the date.

Parameters **attributes** (*dict*) – The JSON object to build this object with.

to_json()

Return the original JSON that was used to construct the object

Abstracts away the pagination of the LabStats API

class labstats_api.paginated_list.**AfterIDPaginatedList**(*content_class*, *requester*, *request_endpoint*, *request_method='GET'*, *first_id=1*, ***kwargs*)

Abstracts the `after_id` type pagination in LabStats, such as that used on `/apps`

Parameters

- **content_class** (*labstats_api.models.LabStatsObject*) – The type of object that this list will contain. All data received from LabStats will be used to create this type of object.
- **requester** (*labstats_api.requester.Requester*) – Requester instance to use to get data into this list
- **request_endpoint** (*str*) – The LabStats API endpoint to get data from. Omit the leading slash. Add a trailing slash. For example, `apps/`
- **request_method** (*str*) – `GET` is the only supported method for now
- **first_id** (*int*) – The exclusive ID of the minimum element that should go in the list. LabStats returns data starting at the nearest ID larger than it.

Additional parameters to the request can be passed as `kwargs`.

Abstractions for communication with the LabStats REST API

class labstats_api.requester.**Requester**(*api_url*, *api_key*, *session*, *timeout=10*)

Abstracts communication with the LabStats REST API.

Parameters

- **api_url** (*str*) – URL to LabStats API starting with `https://`
- **api_key** (*str*) – API authorization key to query the LabStats server with. See API Key Creation for more information.
- **session** – A Requests Session object to use for this instance. See Advanced Usage – Requests for more information.
- **timeout** (*int or tuple*) – A Requests-compatible timeout value to be used on all requests. See Timeouts – Requests for more information.

request_data(*method*, *endpoint*, *parameters=None*, *headers=None*)

Request data from the LabStats API

Parameters

- **method** (*str*) – The method to use. Only `"GET"` is implemented
- **endpoint** (*str*) – The endpoint URL in LabStats, such as `"apps/"`. Do not give a leading slash. Give a trailing slash.
- **parameters** (*dict*) – URL parameters
- **headers** (*dict*) – Extra HTTP headers to send with the request

CHAPTER 4

Indices and tables

- genindex
- modindex
- search

Python Module Index

l

Index

I

K

L

M

N

O

P

R

S

T

U

V