rt_api Documentation

Release 0.0.1

Nicholas Molloy

Jul 14, 2018

Contents:

rt_api is a python client for the Rooster Teeth Api. It allows easy access to resources such as episodes, seasons, shows, and users.

It supports Python 2.7, 3.4, 3.5, 3.6, and 3.7

CHAPTER 1

Quickstart

1.1 Installation

To install rt_api, run:

```
pip install rt_api
```

Alternatively rt_api can be installed from source by cloning the repository and running setuptools:

```
git clone https://github.com/NickMolloy/rt_api
cd rt_api
python setup.py install
```

1.2 Using rt_api

The main entry point for the library is the *Api* class. Instantiating this class will give access to all of the API functionality. For example:

```
from rt_api.api import Api

api = Api()  # Instantiate api. Generates default access token.
latest_episodes = api.episodes()  # Get an iterable of the latest episodes
newest_episode = next(latest_episodes)
print(newest_episode.title)  # Print out episode title
show = newest_episode.show  # Get a reference to the show the episode is from
print(show.name)  # Print out name of the show
```

If you want to be able to perform actions as a specific user, you must first authenticate:

```
from rt_api.api import Api

```

(continues on next page)

(continued from previous page)

```
api = Api()
api.authenticate("myUsername", "myPassword")  # Authenticate as myUsername
```

From this point, all actions performed will be done in the context of that user. For instance, the current user is available through the `me` attribute of the api:

```
my_user = api.me  # Get the user object associated with the authenticated user
my_user.queue  # Get the current user's episode watch list
```

For more information on the available actions, see the *package documentation*, or some *examples*.

1.3 Codebase Overview

`rt_api.api` contains the api itself, which actually performs the api actions.

`rt_api.models` contains the models of the api.

1.3.1 rt_api

rt_api package

rt_api.api module

Module containing the interface to the Api.

Currently access to Episodes, Shows, Seasons, and Users is provided.

Todo:

- Get live videos
- Login/Authenticate with Facebook
- Feed for News
- Feed for Forum topics
- Registration/Sign up
- Signing up for Sponsorship
- Password update and reset
- Changing user profile picture
- Caching responses

class rt_api.api.**Api**(*api_key=None*)

Bases: `object`

Main class of the API.

Create an instance of this to access the api.

__init__(*api_key=None*)

Create an api object.

Parameters **api_key** (*str, optional*) – api key to use. If one is not supplied, a default one will be generated and used.

token

Return the token currently in use by the api.

Returns Token currently in use by this instance.

Return type str

authenticate (*username*, *password*)

Authenticate to the API using a username and password.

The token retrieved will be used for future API requests, and will persist only until the api object is destroyed. If you wish to use this token in future sessions, you should save the token and use it again when creating an api object.

The token retrieved is linked to the credentials used to authenticate. Using the token in the api mean actions performed will be done as the user with those credentials.

Parameters

- **username** (*str*) – Username of Rooster Teeth account.
- **password** (*str*) – Password of Rooster Teeth account.

Returns Token retrieved from API on successful authentication.

Return type str

Raises *AuthenticationError* – if authentication fails.

me

Access the *User* object for the authenticated user.

If not authenticated as a user, returns None.

Returns The user object corresponding to authenticated user or None.

Return type *User*

episode (*episode_id*)

Retrieve the episode corresponding to the specified id.

Parameters **episode_id** (*int*) – ID of the episode to retrieve.

Returns Episode instance.

Return type *Episode*

episodes (*site=None*, *page=1*, *count=20*)

Get latest episodes from feed.

Parameters

- **site** (*str, optional*) – If specified, only episodes from this site will be returned.
- **page** (*int*) – The page to start from (Default value = 1).
- **count** (*int*) – Number of Episodes per page (Default value = 20).

Returns An iterable collection of *Episodes* from 'latest' feed.

Return type iterable

season (*season_id*)

Retrieve the season corresponding to the specified id.

Parameters **season_id** (*int*) – ID of the season to retrieve.

Returns Season instance.

Return type *Season*

season_episodes (*season_id*)

Retrieve the episodes that belong to the season with the specified id.

Parameters **season_id** (*int*) – ID of the season.

Returns A list of `Episode` objects.

Return type list

show_seasons (*show_id*)

Get the seasons belonging to show with specified ID.

Parameters **show_id** (*int*) – ID of the show.

Returns A list of `Season` objects.

Return type list

show (*show_id*)

Return show with given id.

Parameters **show_id** (*int*) – ID of the show to retrieve.

Returns Show instance.

Return type *Show*

shows (*site=None*, *page=1*, *count=20*)

Return an iterable feed of `Shows`.

This will return an iterable, which starts at the specified page, and can be iterated over to retrieve all shows onwards.

Under the hood, as this is iterated over, new pages are fetched from the API. Therefore, the size of `count` will dictate the delay this causes.

A larger `count` means larger delay, but fewer total number of pages will need to be fetched.

Parameters

- **site** (*str*) – Only return shows from specified site, or all sites if None.
- **page** (*int*) – The page to start from (Default value = 1).
- **count** (*int*) – Number of Shows per page (Default value = 20).

Returns An iterable collection of `Shows`.

Return type iterable

Example:

```
r = rt_api()
show_feed = r.shows(site="theKnow")
for show in show_feed:
    print(show)
```

user (*user_id*)

Retrieve the User with the specified id.

Parameters **user_id** (*int*) – ID of the user to retrieve.

Returns User instance.

Return type *User*

update_user_details(***kwargs*)

Update the details of the user with the specified id.

You must be authenticated as the user to be updated. Attributes should be specified as keyword arguments.

Possible keyword arguments: displayTitle, name, sex, location, occupation, about

Note: All attributes will be updated. If an attribute is not specified, the remote end assumes it to be empty and sets it as such.

Parameters **user_id** (*int*) – ID of the user to update.

Raises *NotAuthenticatedError* – if not currently authenticated as a user, or this is attempted on a user not authenticated as.

user_queue(*user_id*, *page=1*, *count=20*)

Retrieve the episodes in specified user's queue.

Parameters

- **user_id** (*int*) – The ID of the user to get the queue of.
- **page** (*int*) – The page to start from (Default value = 1).
- **count** (*int*) – Number of Episodes per page (Default value = 20).

Returns Iterable of *Episode* instances.

Return type iterable

add_episode_to_queue(***kwargs*)

Add specified episode to current user's queue.

Parameters **episode_id** (*int*) – ID of the episode to add to user's queue.

Returns Success message from API or None.

Return type str

Raises *NotAuthenticatedError* – if not currently authenticated as a user.

remove_episode_from_queue(***kwargs*)

Remove specified episode from current user's queue.

Parameters **episode_id** (*int*) – ID of the episode to remove from user's queue.

Returns Success message from API or None.

Return type str

Raises *NotAuthenticatedError* – if not currently authenticated as a user.

mark_episode_watched(***kwargs*)

Mark the specified episode as having been watched by the current user.

Parameters **episode_id** (*int*) – ID of the episode to mark as having been watched.

search(*query*, *include=None*)

Perform a search for the specified query.

Currently only supports searching for Episodes, Shows, and Users. Unfortunately, the Api only returns up to 10 of each resource type.

Parameters

- **query** (*str*) – The value to search for.
- **include** (*list, optional*) – A list of types to include in the results (Default value = None). If `include` is specified, only objects of those types will be returned in the results.

Example

Search for "funny", only in shows and episodes.

```
search("funny", include=[rt_api.models.Show, rt_api.models.Episode])
```

Returns The search results.

Return type list

exception `rt_api.api.`**`AuthenticationError`**

Bases: `exceptions.Exception`

Raised when an error is encountered while performing authentication.

exception `rt_api.api.`**`NotAuthenticatedError`**

Bases: `exceptions.Exception`

Raised if an action requiring authentication is attempted but no account is authenticated.

rt_api.models module

Module containing the models used for the API.

The classes here can be used independently of the API.

When instantiating an *Episode*, *Show*, *Season*, or *User* the `api` parameter is optional. If it is specified, that object will be used to make calls to the API when needed.

For example, *Episode.season* would make a call to the API. If `api` is not specified, `NotImplementedError` will be raised if a method that needs the api is called.

class `rt_api.models.`**`ApiObject`**

Bases: `object`

Base class for resources available from the API.

Resources such as Episodes, Seasons, Shows, and Users should inherit from this class.

`__init__`()

x.__init__(...) initializes x; see help(type(x)) for signature

`get_thumbnail`(*quality*)

Return the url of the thumbnail of the resource at specified quality.

Parameters **quality** (*str*) – possible values are (in order from smallest to largest): "tb", "sm", "md", "lg".

Returns URL of the thumbnail or `None` if thumbnail not available at specified quality.

Return type str

`__eq__`(*other*)
Define equality of two API objects as having the same type and attributes.

`class rt_api.models.`**`Show`**(*show_json*, *api=None*)
Bases: *`rt_api.models.ApiObject`*

Class representing a Show.

`id_`
int – Identifier of the show.

`name`
str – The name of the show.

`summary`
str – Summary of the show.

`thumbnail`
str – URL of the default sized thumbnail of the Show.

`cover_picture`
str – URL of the default sized cover_picture of the Show.

`season_count`
int – Number of seasons this show has.

`seasons`
list – All of the *`Seasons`* of the show.

`episodes`
list – All of the *`Episodes`* of the show. This is equivalent to iterating over the show's seasons, and then iterating over each season's episodes.

`canonical_url`
str – URL of the show on the website. e.g. http://roosterteeth.com/show/on-the-spot

`__init__`(*show_json*, *api=None*)
Take in a JSON representation of a show and convert it into a Show Object.

Parameters

- **`show_json`** (*`json`*) – JSON representation of a show resource.
- **`api`** (*`object, optional`*) – Object that implements the API (see `rt_api`). This will be used to make calls to the API when needed.

`class rt_api.models.`**`Season`**(*season_json*, *api=None*)
Bases: *`rt_api.models.ApiObject`*

Class representing a Season.

`id_`
int – Identifier of this season.

`number`
int – The Season number.

`title`
str – Title of the season.

`description`
str – Description of the season.

episodes
list – All of the `Episodes` of this season.

show_name
str – Name of the `Show` this season belongs to.

show_id
int – Identifier of the `Show` this season belongs to.

show
`Show` – The `Show` this season belongs to.

__init__(*season_json*, *api=None*)
Take in a JSON representation of a season and convert it into a Season Object.

Parameters

- **season_json** (*json*) – JSON representation of a season resource.
- **api** (*object, optional*) – Object that implements the API (see `rt_api`). This will be used to make calls to the API when needed.

class rt_api.models.**Episode**(*episode_json*, *api=None*)
Bases: `rt_api.models.ApiObject`

Class representing an Episode.

id_
int – Identifier of the episode.

title
str – Title of the episode.

number
int – The number of this episode.

caption
str – Caption of the episode.

description
str – Description of the episode.

is_sponsor_only
bool – Indicates whether or not the episode is only available to sponsor members.

video
`Video` – The `Video` object associated with the episode or `None` if no video is available. If the video is sponsor only and a sponsor user is not currently authenticated then `video` will be `None`.

length
int – Length of the episode in seconds.

thumbnail
str – URL of the default sized thumbnail of the episode.

is_watched
bool – Indicates whether the current user has watched this episode. Will default to `False` if not currently authenticated as a user. Note: It seems that the API doesn't actually set this value. Even if you mark an episode as watched, the API returns it as still being unwatched.

links
list – List of `Link` objects associated with the episode.

audio
Audio – The *Audio* object associated with the episode or `None`.

canonical_url
str – URL to the episode on the website. e.g. ‘http://roosterteeth.com/episode/the-know-game-news-season-1-crytek-shuts-down-studios’.

site
str – The ‘site’ this episode belongs to. e.g. ‘theKnow’.

show_name
str – The name of the *Show* this episode belongs to.

show_id
int – Identifier of the *Show* this episode belongs to.

show
Show – The *Show* object this episode belongs to.

season_id
int – Identifier of the *Season* this episode belongs to.

season
Season – The *Season* object this episode belongs to.

__init__(*episode_json*, *api=None*)
Take in a JSON representation of an episode and convert it into an Episode Object.

Parameters

- **episode_json** (*json*) – JSON representation of an episode resource.
- **api** (*object, optional*) – Object that implements the API (see `rt_api`). This will be used to make calls to the API when needed.

mark_as_watched(***kwargs*)
Mark this episode as watched by current user.

Note: It seems that the API doesn’t actually set this value. Even if an episode is marked as watched, the API returns it as still being unwatched.

Raises *NotAuthenticatedError* – if not currently authenticated as a user.

add_to_queue(***kwargs*)
Add this episode to current user’s queue.

Raises *NotAuthenticatedError* – if not currently authenticated as a user.

remove_from_queue(***kwargs*)
Remove this episode from current user’s queue.

Raises *NotAuthenticatedError* – if not currently authenticated as a user.

class rt_api.models.**User**(*user_json*, *api=None*)
Bases: *rt_api.models.ApiObject*

Represents a User.

id_
int – Identifier of the user.

username
str – Username of the user.

name
str – Name of the user.

is_sponsor
bool – `True` if user is a sponsor, else `False`.

location
str – Location of the user.

occupation
str – Occupation of the user.

about
str – About the user.

sex
str – Sex of the user.

display_title
str – Display title of the user.

has_used_trial
bool – `True` if user has previously used free trial.

canonical_url
str – URL of the user on the website.

queue
list – List of *Episodes* the user has added to their queue.

thumbnail
str – URL of the default sized thumbnail of the user.

__init__(*user_json*, *api=None*)
Take in a JSON representation of a user and convert it into a User Object.

Parameters

- **user_json** (*json*) – JSON representation of a user resource.
- **api** (*object, optional*) – Object that implements the API (see `rt_api`). This will be used to make calls to the API when needed.

update(***kwargs*)
Update user details on remote end.

Should be called after a user's attributes are changed if the changes should persist.

Raises *NotAuthenticatedError* – if not currently authenticated as a user, or this is attempted on a user that we are not authenticated as.

Example

```
>>> my_user.name = "NewName"
>>> my_user.update()
```

class rt_api.models.**Video**(*url*, *type_*)
Bases: `object`

Encapsulates a video resource.

A Video represents a HLS video resource. It is usually available in multiple qualities.

Todo: Add support for live videos.

`__init__`(*url*, *type_*)

Create a video resource.

Parameters

- **`url`** (*`str`*) – URL to the index m3u8 file of the video.
- **`type`** (*`str`*) – The type of video (e.g. "cdn" or "ustream").

Example

Video("http://example.com/someVideo/index.m3u8", "cdn")

`available_qualities`

Return a list of the qualities the video is available in.

Returns A list of the qualities (as resolutions) the video is available in.

Return type list

Example

```
>>> some_video.available_qualities
["720P", "480P"]
```

`get_quality`(*quality=None*)

Return the chosen quality if specified.

If quality is specified, that will be returned, otherwise: If a quality has previously been selected return that, otherwise: Return default quality (720P).

Parameters **`quality`** (*`str`*) – Resolution of video. Should be in format <resolution>P

Returns The full URL of the video at the specified quality.

Return type str

Example

```
>>> some_video.get_quality()
'http://wpc.1765A.taucdn.net/801765A/video/uploads/videos/c762632a-b8de-4859-
↪962d-607d8e77ccc4/NewHLS-720P.m3u8'
```

```
>>> some_video.get_quality(quality="360P")
'http://wpc.1765A.taucdn.net/801765A/video/uploads/videos/c762632a-b8de-4859-
↪962d-607d8e77ccc4/NewHLS-360P.m3u8'
```

set_selected_quality(*quality*)

Set the preferred quality of this video.

Subsequent calls to *get_quality()* will return this selected quality (if no quality argument is given to it).

Parameters **quality** (*str*) – Resolution of video. Should be in format <resolution>P

Raises KeyError – if the specified quality is not available.

Example

```
>>> set_selected_quality("480P")
>>> get_quality()
'http://wpc.1765A.taucdn.net/801765A/video/uploads/videos/c762632a-b8de-4859-
↪962d-607d8e77ccc4/NewHLS-480P.m3u8'
```

class rt_api.models.**Audio**(*url*, *container*)

Bases: object

Encapsulates an audio resource.

url

str – URL of the audio resource.

container

str – The container format of the audio resource.

__init__(*url*, *container*)

Create an audio resource.

Parameters

- **url** (*str*) – URL of the audio resource.
- **container** (*str*) – The container format of the audio resource e.g “mp3”.

class rt_api.models.**Link**(*url*, *title*, *thumbnail=None*)

Bases: object

Encapsulates a link resource.

url

str – URL of the link resource.

title

str – The title of the Link resource.

thumbnail

Thumbnail – The Thumbnail object that represents this Link.

__init__(*url*, *title*, *thumbnail=None*)

Create a Link resource.

Parameters

- **url** (*str*) – URL of the Link resource.
- **title** (*str*) – The title of the Link resource.
- **thumbnail** (*Thumbnail*, optional) – The Thumbnail object that represents this Link. (Default value = None).

class rt_api.models.**Thumbnail**(*thumbnail_json*)

Bases: dict

Represents the available thumbnails of an API resource.

The keys of the dictionary are the qualities the thumbnail is available in: “original”, “tb”, “sm”, “md”, “lg”. The corresponding values are the URL of the thumbnail at that quality.

qualities = ['tb', 'sm', 'md', 'original']

Possible qualities of a thumbnail in ascending order of size.

__init__(*thumbnail_json*)

Create a Thumbnail resource.

Parameters **thumbnail_json** (*json*) – JSON representation of a thumbnail.

class rt_api.models.**MediaGroup**(*media_json*)

Bases: object

Class that encapsulates all the media items of an Episode.

An episode may have multiple videos, audio items, and links associated with it.

videos

list – A list of all the *Video* items of an episode.

audio

list – A list of all the *Audio* items of an episode.

links

list – A list of all the *Link* items of an episode.

__init__(*media_json*)

Create a MediaGroup resource.

Parameters **media_json** (*json*) – JSON representation of the media resource.

1.3.2 Example Usage

```
from rt_api.api import Api
import itertools

# Instantiate Api
api = Api()
# Get an iterator for the episode feed for Achievement Hunter
episode_feed = api.episodes(site="AchievementHunter")
# Slice first 5 episodes from feed into a list
latest_episodes = list(itertools.islice(episode_feed, 4))
# Iterate over these latest episodes
for episode in latest_episodes:
    print("{0} - {1}".format(episode.show_name, episode.title))
    print("\t{0} seconds long".format(episode.length))
    if episode.is_sponsor_only:
        print("\tIs only available to sponsors")

# Sign in as a user
api.authenticate("someUsername", "somePassword")
# Add episode to our queue
episode = latest_episodes[0]
episode.add_to_queue()
```

(continues on next page)

(continued from previous page)

```
# Get a reference to our user
me = api.me
print("My username is {0}, my name is {1}".format(me.username, me.name))
print("I {0} a sponsor".format("am" if me.is_sponsor else "am not"))
# Change our name
me.name = "New Name"
me.update()
print("Now my name is {0}".format(me.name))
```

Iterables

Some methods of `Api` return an iterable collection of a resource e.g `episodes()`, `shows()`, and `user_queue()`. The iterable returned starts at the specified page, and can be used to retrieve all items onwards.

For example, if `page` is set to 2 then the iterable returned will allow iteration over all items from item 21 onwards (The default value of `count` is 20).

Example:

```
from rt_api.api import Api

api = Api()
show_feed = api.shows(page=5)
# Can iterate over all shows, starting at the 81st item from the feed
for show in show feed:
    # Do something with show
    pass
```

CHAPTER 2

Indices and tables

- genindex
- modindex
- search

Python Module Index

Index