python-twitch-client Documentation

Release 0.1

Tomaz Sifrer

Mar 16, 2021

Contents

An easy to use Python library for accessing the Twitch API

Warning: This documentation is a work in progress

Note: `python-twitch-client` currently supports Helix API and Twitch API v5.

Helix API integration is a work in progress and some endpoints might be missing.

If you find a missing endpoint or a bug please raise an issue or contribute and open a pull request.

CHAPTER 1

Installation

You can install `python-twitch-client` with `pip`:

```
$ pip install python-twitch-client
```

`python-twitch-client` is currently only tested and confirmed working on Linux and Mac. If you're on a Windows machine and getting any bugs, please open a bug and help us find a solution.

CHAPTER 2

Authentication

Before you can use Twitch API you need to get the client ID. To get one, you should follow the steps on Twitch Authentication page.

Some of the endpoints also require OAuth token. To get one for testing purposes, you can use the free tokengen tool or use TwitchHelix's `get_oauth` method.

There are two ways to pass credentials into the TwitchClient. The first and easiest way is to just pass the credentials as an argument:

```
client = TwitchClient(client_id='<my client id>', oauth_token='<my oauth token>')
```

Other option is to create a config file *~/.twitch.cfg* which is a text file formatted as .ini configuration file.

An example of the config file might look like:

```
[Credentials]
client_id = <my client id>
oauth_token = <my oauth token>
```

Note: You only need to provide `oauth_token` if you're calling endpoints that need it.

If you call functions that require `oauth_token` and you did not provide it, functions will raise `TwitchAuthException` exception.

2.1 Contents:

2.1.1 Basic Usage

The `python-twitch-client` allows you to easily access to Twitch API endpoints.

This package is a modular wrapper designed to make Twitch API calls simpler and easier for you to use. Provided below are examples of how to interact with commonly used API endpoints, but this is by no means a complete list.

Getting a channel by ID

```
from twitch import TwitchClient

client = TwitchClient(client_id='<my client id>')
channel = client.channels.get_by_id(44322889)

print(channel.id)
print(channel.name)
print(channel.display_name)
```

2.1.2 Twitch Helix

Helix is the latest version of Twitch API

class twitch.helix.**TwitchHelix**(*client_id=None*, *oauth_token=None* *client_secret=None*, *scopes=None*)

This class provides methods for easy access to Twitch Helix API.

Parameters

- **client_id** (*string*) – Client ID you get from your registered app on Twitch
- **oauth_token** (*string*) – OAuth token, if you already have it, otherwise use `client_secret` and `scopes` then call `get_oauth` to generate it
- **client_secret** (*string*) – Client secret. Only used by `get_oauth` and should only be present if oauth_token is not set
- **scopes** (*string*) – Twitch scopes that we want the OAuth token to have. Only used by `get_oauth` and should only be present if oauth_token is not set

Basic usage with oauth_token set:

```
import twitch
client = twitch.TwitchHelix(client_id='<client_id>', oauth_token='<oauth_token>')
client.get_streams()
```

Basic usage with fetching the oauth token on initialization:

```
import twitch
client = twitch.TwitchHelix(client_id='<client_id>', client_secret='<client_
↪secret>', scopes=[twitch.constants.OAUTH_SCOPE_ANALYTICS_READ_EXTENSIONS])
client.get_oauth()
client.get_streams()
```

classmethod get_oauth(*clip_id*)

Gets access token with access to the requested scopes and stores the token on the class for future usage

classmethod get_clips(*broadcaster_id=None*, *game_id=None*, *clip_ids=None*, *after=None*, *before=None*, *started_at=None*, *ended_at=None*, *page_size=20*)

Gets clip information by clip ID (one or more), broadcaster ID (one only), or game ID (one only).

Parameters

- **broadcaster_id** (*string*) – Broadcaster ID for whom clips are returned. The number of clips returned is determined by the `page_size` parameter (Default: 20 Max: 100). Results are ordered by view count.
- **game_id** (*string*) – Game ID for which clips are returned. The number of clips returned is determined by the `page_size` parameter (Default: 20 Max: 100). Results are ordered by view count.
- **clip_ids** (*list*) – List of clip IDS being queried. Limit: 100.
- **after (optional)** (*string*) – Cursor for forward pagination: tells the server where to start fetching the next set of results, in a multi-page response.
- **before (optional)** (*string*) – Cursor for backward pagination: tells the server where to start fetching the next set of results, in a multi-page response.
- **started_at (optional)** (*string*) – Starting date/time for returned clips, in RFC3339 format. (The seconds value is ignored.) If this is specified, ended_at also should be specified; otherwise, the ended_at date/time will be 1 week after the started_at value.
- **ended_at (optional)** (*string*) – Ending date/time for returned clips, in RFC3339 format. (Note that the seconds value is ignored.) If this is specified, started_at also must be specified; otherwise, the time period is ignored.
- **page_size (optional)** (*integer*) – Number of objects returned in one call. Maximum: 100. Default: 20.

Returns `APICursor` if `broadcaster_id` or `game_ids` are provided, returns list of `Clip` objects instead.

For response fields of `get_clips` and official documentation check Twitch Helix Get Clips.

classmethod get_games(*game_ids=None*, *names=None*)

Gets game information by game ID or name.

Parameters

- **game_ids** (*list*) – List of Game IDs. At most 100 id values can be specified.
- **names** (*list*) – List of Game names. The name must be an exact match. For instance, "Pokemon" will not return a list of Pokemon games; instead, query the specific Pokemon game(s) in which you are interested. At most 100 name values can be specified.

Returns `APICursor` containing `Game` objects

For response fields of `get_games` and official documentation check Twitch Helix Get Games.

classmethod get_streams(*after=None*, *before=None*, *community_ids=None*, *page_size=20*, *game_ids=None*, *languages=None*, *stream_type=None*, *user_ids=None*, *user_logins=None*)

Gets information about active streams. Streams are returned sorted by number of current viewers, in descending order. Across multiple pages of results, there may be duplicate or missing streams, as viewers join and leave streams.

Parameters

- **after** (*string*) – Cursor for forward pagination: tells the server where to start fetching the next set of results, in a multi-page response.
- **before** (*string*) – Cursor for backward pagination: tells the server where to start fetching the next set of results, in a multi-page response.
- **community_ids** (*list*) – Returns streams in a specified community ID. You can specify up to 100 IDs.

- **page_size** (*integer*) – Number of objects returned in one call. Maximum: 100. Default: 20.
- **game_ids** (*list*) – Returns streams broadcasting a specified game ID. You can specify up to 100 IDs.
- **languages** (*list*) – Stream language. You can specify up to 100 languages
- **user_ids** (*list*) – Returns streams broadcast by one or more specified user IDs. You can specify up to 100 IDs.
- **user_logins** (*list*) – Returns streams broadcast by one or more specified user login names. You can specify up to 100 names.

Returns `APICursor` containing `Stream` objects

For response fields of `get_streams` and official documentation check Twitch Helix Get Streams.

classmethod get_top_games(*after=None*, *before=None*, *page_size=20*)

Gets games sorted by number of current viewers on Twitch, most popular first.

Parameters

- **after** (*string*) – Cursor for forward pagination: tells the server where to start fetching the next set of results, in a multi-page response.
- **before** (*string*) – Cursor for backward pagination: tells the server where to start fetching the next set of results, in a multi-page response.
- **page_size** (*integer*) – Number of objects returned in one call. Maximum: 100. Default: 20.

Returns `APICursor` containing `Game` objects

For response fields of `get_top_games` and official documentation check Twitch Helix Get Top Games.

classmethod get_videos(*video_ids=None*, *user_id=None*, *game_id=None*, *after=None*, *before=None*, *page_size=20*, *language=None*, *period=None*, *sort=None*, *video_type=None*)

Gets video information by video ID (one or more), user ID (one only), or game ID (one only).

Parameters

- **video_ids** (*list*) – List of Video IDs. Limit: 100. If this is specified, you cannot use any of the optional query string parameters below.
- **user_id** (*string*) – User ID who owns the videos.
- **game_id** (*int*) – Game ID of the videos.
- **after (optional)** (*string*) – Cursor for forward pagination: tells the server where to start fetching the next set of results, in a multi-page response.
- **before (optional)** (*string*) – Cursor for backward pagination: tells the server where to start fetching the next set of results, in a multi-page response.
- **page_size (optional)** (*integer*) – Number of objects returned in one call. Maximum: 100. Default: 20.
- **language (optional)** (*string*) – Language of the video being queried.
- **period (optional)** (*string*) – Period during which the video was created. Valid values: `VIDEO_PERIODS`. Default: `VIDEO_PERIOD_ALL`
- **sort (optional)** (*string*) – Sort order of the videos. Valid values: `VIDEO_SORTS`. Default: `VIDEO_SORT_TIME`

- **type (optional)** (*string*) – Type of video. Valid values: `VIDEO_TYPES`. Default: `VIDEO_TYPE_ALL`

Returns `APICursor` if `user_id` or `game_id` are provided, returns list of `Video` objects instead.

For response fields of `get_videos` and official documentation check Twitch Helix Get Videos.

classmethod get_streams_metadata(*after=None*, *before=None*, *community_ids=None*, *page_size=20*, *game_ids=None*, *languages=None*, *stream_type=None*, *user_ids=None*, *user_logins=None*)

Gets metadata information about active streams playing Overwatch or Hearthstone. Streams are sorted by number of current viewers, in descending order. Across multiple pages of results, there may be duplicate or missing streams, as viewers join and leave streams.

Parameters

- **after** (*string*) – Cursor for forward pagination: tells the server where to start fetching the next set of results, in a multi-page response.
- **before** (*string*) – Cursor for backward pagination: tells the server where to start fetching the next set of results, in a multi-page response.
- **community_ids** (*list*) – Returns streams in a specified community ID. You can specify up to 100 IDs.
- **page_size** (*integer*) – Number of objects returned in one call. Maximum: 100. Default: 20.
- **game_ids** (*list*) – Returns streams broadcasting a specified game ID. You can specify up to 100 IDs.
- **languages** (*list*) – Stream language. You can specify up to 100 languages
- **user_ids** (*list*) – Returns streams broadcast by one or more specified user IDs. You can specify up to 100 IDs.
- **user_logins** (*list*) – Returns streams broadcast by one or more specified user login names. You can specify up to 100 names.

Returns `APICursor` containing `StreamMetadata` objects

For response fields of `get_streams` and official documentation check Twitch Helix Get Streams Metadata.

classmethod get_user_follows(*after=None*, *page_size=20*, *from_id=None*, *to_id=None*)

Gets information on follow relationships between two Twitch users. Information returned is sorted in order, most recent follow first. This can return information like "who is lirik following," "who is following lirik," or "is user X following user Y."

Parameters

- **after** (*string*) – Cursor for forward pagination: tells the server where to start fetching the next set of results, in a multi-page response.
- **page_size** (*integer*) – Number of objects returned in one call. Maximum: 100. Default: 20.
- **from_id** (*list*) – User ID. The request returns information about users who are being followed by the `from_id` user.
- **to_id** (*list*) – User ID. The request returns information about users who are following the `to_id` user.

Returns `APICursor` containing `Follow` objects

For response fields of `get_streams` and official documentation check Twitch Helix Get Users Follows.

2.1.3 Twitch v5

Clips

class `twitch.api.clips.`**Clips**

This class provides methods for easy access to Twitch Clips API.

classmethod get_by_slug(*slug*)

Gets a clip object based on the slug provided

Parameters **slug** (*string*) – Twitch Slug.

classmethod get_top(*channel*, *cursor*, *game*, *language*, *limit*, *period*, *trending*)

Gets all clips emoticons in one or more specified sets.

Parameters

- **channel** (*string*) – Channel name. If this is specified, top clips for only this channel are returned; otherwise, top clips for all channels are returned. If both channel and game are specified, game is ignored.
- **cursor** (*string*) – Tells the server where to start fetching the next set of results, in a multi-page response.
- **game** (*string*) – Game name. (Game names can be retrieved with the Search Games endpoint.) If this is specified, top clips for only this game are returned; otherwise, top clips for all games are returned. If both channel and game are specified, game is ignored.
- **language** (*string*) – Comma-separated list of languages, which constrains the languages of videos returned. Examples: es, en,es,th. If no language is specified, all languages are returned.
- **limit** (*int*) – Maximum number of most-recent objects to return. Default: 10. Maximum: 100.
- **period** (*string*) – The window of time to search for clips. Valid values: day, week, month, all. Default: week.
- **trending** (*boolean*) – If True, the clips returned are ordered by popularity; otherwise, by viewcount. Default: False.

classmethod followed()

Gets top clips for games followed by the user identified by OAuth token. Results are ordered by popularity.

Parameters

- **limit** (*int*) – Maximum number of most-recent objects to return. Default: 10. Maximum: 100.
- **cursor** (*string*) – Tells the server where to start fetching the next set of results, in a multi-page response.
- **trending** (*boolean*) – If true, the clips returned are ordered by popularity; otherwise, by viewcount. Default: false.

Chat

class twitch.api.chat.**Chat**

This class provides methods for easy access to Twitch Chat API.

classmethod get_badges_by_channel(*channel_id*)

Gets a list of badges that can be used in chat for a specified channel.

Parameters **channel_id** (*string*) – Channel ID.

classmethod get_emoticons_by_set(*emotesets*)

Gets all chat emoticons in one or more specified sets.

Parameters **emotesets** (*list*) – List of emoticon sets to be returned.

classmethod get_all_emoticons()

Gets all chat emoticons.

```
>>> from twitch import TwitchClient
>>> client = TwitchClient('<my client id>')
>>> emotes = client.chat.get_all_emoticons()
```

Channel feed

class twitch.api.channel_feed.**ChannelFeed**

This class provides methods for easy access to Twitch Channel Feed API.

classmethod get_posts(*channel_id*, *limit*, *cursor*, *comments*)

Gets posts from a specified channel feed.

Parameters

- **channel_id** (*string*) – Channel ID
- **limit** (*int*) – Maximum number of objects to return. Default 10. Maximum 100.
- **cursor** (*string*) – Cursor of the next page
- **comments** (*int*) – Number of comments to return. Default 5. Maximum 5.

classmethod get_post(*channel_id*, *post_id*, *comments*)

Gets a specified post from a specified channel feed.

Parameters

- **channel_id** (*string*) – Channel ID
- **post_id** (*string*) – Post ID
- **comments** (*int*) – Number of comments to return. Default 5. Maximum 5.

```
>>> from twitch import TwitchClient
>>> client = TwitchClient('<my client id>')
>>> post = client.channel_feed.get_post('12345', '12345', comments=0)
```

classmethod create_post(*channel_id*, *content*, *share*)

Creates a post in a specified channel feed.

Parameters

- **channel_id** (*string*) – Channel ID
- **content** (*string*) – Content of the post

- **share** (*boolean*) – When set to true, the post is shared on the channel's Twitter feed.

classmethod delete_post(*channel_id*, *post_id*)
Deletes a specified post in a specified channel feed.

Parameters

- **channel_id** (*string*) – Channel ID
- **post_id** (*string*) – Post ID

classmethod create_reaction_to_post(*channel_id*, *post_id*, *emote_id*)
Creates a reaction to a specified post in a specified channel feed.

Parameters

- **channel_id** (*string*) – Channel ID
- **post_id** (*string*) – Post ID
- **emote_id** (*string*) – Emote ID

classmethod delete_reaction_to_post(*channel_id*, *post_id*, *emote_id*)
Deletes a specified reaction to a specified post in a specified channel feed.

Parameters

- **channel_id** (*string*) – Channel ID
- **post_id** (*string*) – Post ID
- **emote_id** (*string*) – Emote ID

classmethod get_post_comments(*channel_id*, *post_id*, *limit*, *cursor*)
Gets all comments on a specified post in a specified channel feed.

Parameters

- **channel_id** (*string*) – Channel ID
- **post_id** (*string*) – Post ID
- **limit** (*int*) – Maximum number of objects to return. Default 10. Maximum 100.
- **cursor** (*string*) – Cursor of the next page

classmethod create_post_comment(*channel_id*, *post_id*, *content*)
Creates a comment to a specified post in a specified channel feed.

Parameters

- **channel_id** (*string*) – Channel ID
- **post_id** (*string*) – Post ID
- **content** (*string*) – Content of the comment

classmethod delete_post_comment(*channel_id*, *post_id*, *comment_id*)
Deletes a specified comment on a specified post in a specified channel feed.

Parameters

- **channel_id** (*string*) – Channel ID
- **post_id** (*string*) – Post ID
- **comment_id** (*string*) – Comment ID

classmethod create_reaction_to_comment(*channel_id*, *post_id*, *comment_id*, *emote_id*)
Creates a reaction to a specified comment on a specified post in a specified channel feed.

Parameters

- **channel_id** (*string*) – Channel ID
- **post_id** (*string*) – Post ID
- **comment_id** (*string*) – Comment ID
- **emote_id** (*string*) – Emote ID

classmethod delete_reaction_to_comment(*channel_id*, *post_id*, *comment_id*, *emote_id*)
Deletes a reaction to a specified comment on a specified post in a specified channel feed.

Parameters

- **channel_id** (*string*) – Channel ID
- **post_id** (*string*) – Post ID
- **comment_id** (*string*) – Comment ID
- **emote_id** (*string*) – Emote ID

Channels

class twitch.api.channels.**Channels**
This class provides methods for easy access to Twitch Channels API.

classmethod get()
Gets a channel object based on the OAuth token.

```
>>> from twitch import TwitchClient
>>> client = TwitchClient('<my client id>', '<my oauth token>')
>>> channel = client.channels.get()
```

classmethod get_by_id(*channel_id*)
Gets a specified channel object.

Parameters **channel_id** (*string*) – Channel ID

classmethod update(*channel_id*, *status*, *game*, *delay*, *channel_feed_enabled*)
Updates specified properties of a specified channel.

Parameters

- **channel_id** (*string*) – Channel ID
- **status** (*string*) – Description of the broadcaster's status.
- **game** (*string*) – Name of game.
- **delay** (*int*) – Channel delay, in seconds.
- **channel_feed_enabled** (*boolean*) – If true, the channel's feed is turned on.

classmethod get_editors(*channel_id*)
Gets a list of users who are editors for a specified channel.

Parameters **channel_id** (*string*) – Channel ID

classmethod get_followers(*channel_id*, *limit*, *offset*, *cursor*, *direction*)
Gets a list of users who follow a specified channel.

Parameters

- **channel_id** (*string*) – Channel ID
- **limit** (*int*) – Maximum number of objects to return. Default 25. Maximum 100.
- **offset** (*int*) – Object offset for pagination of result. Default 0.
- **cursor** (*string*) – Cursor of the next page.
- **direction** (*string*) – Direction of sorting.

classmethod get_teams(*channel_id*)
Gets a list of teams to which a specified channel belongs.

Parameters **channel_id** (*string*) – Channel ID

classmethod get_subscribers(*channel_id*, *limit*, *offset*, *direction*)
Gets a list of users subscribed to a specified channel.

Parameters

- **channel_id** (*string*) – Channel ID
- **limit** (*int*) – Maximum number of objects to return. Default 25. Maximum 100.
- **offset** (*int*) – Object offset for pagination of result. Default 0.
- **direction** (*string*) – Direction of sorting.

classmethod check_subscription_by_user(*channel_id*, *user_id*)
Checks if a specified channel has a specified user subscribed to it.

Parameters

- **channel_id** (*string*) – Channel ID
- **user_id** (*string*) – User ID

classmethod get_videos(*channel_id*, *limit*, *offset*, *broadcast_type*, *language*, *sort*)
Gets a list of videos from a specified channel.

Parameters

- **channel_id** (*string*) – Channel ID
- **limit** (*int*) – Maximum number of objects to return. Default 10. Maximum 100.
- **offset** (*int*) – Object offset for pagination of result. Default 0.
- **broadcast_type** (*string*) – Constrains the type of videos returned.
- **language** (*string*) – Constrains the language of the videos that are returned.
- **sort** (*string*) – Sorting order of the returned objects.

classmethod start_commercial(*channel_id*, *duration*)
Starts a commercial (advertisement) on a specified channel.

Parameters

- **channel_id** (*string*) – Channel ID
- **duration** (*string*) – Duration of the commercial in seconds. Default 30.

classmethod reset_stream_key(*channel_id*)
Deletes the stream key for a specified channel. Stream key is automatically reset.

Parameters **channel_id** (*string*) – Channel ID

classmethod get_community(*channel_id*)
Gets the community for a specified channel.

Parameters **channel_id** (*string*) – Channel ID

classmethod set_community(*channel_id*, *community_id*)
Sets a specified channel to be in a specified community.

Parameters

- **channel_id** (*string*) – Channel ID
- **community_id** (*string*) – Community ID

classmethod delete_from_community(*channel_id*)
Deletes a specified channel from its community.

Parameters **channel_id** (*string*) – Channel ID

Collections

class twitch.api.collections.**Collections**
This class provides methods for easy access to Twitch Collections API.

classmethod get_metadata(*collection_id*)
Gets summary information about a specified collection.

Parameters **collection_id** (*string*) – Collection ID

```
>>> from twitch import TwitchClient
>>> client = TwitchClient('<my client id>')
>>> collection = client.collections.get_metadata('12345')
```

classmethod get(*collection_id*, *include_all_items*)
Gets all items in a specified collection.

Parameters

- **collection_id** (*string*) – Collection ID
- **include_all_items** (*boolean*) – If True, unwatchable VODs are included in the response. Default: false.

classmethod get_by_channel(*channel_id*, *limit*, *cursor*, *containig_item*)
Gets all collections owned by a specified channel.

Parameters

- **channel_id** (*string*) – Channel ID
- **limit** (*int*) – Maximum number of objects to return. Default 10. Maximum 100.
- **cursor** (*string*) – Cursor of the next page
- **containig_item** (*string*) – Returns only collections containing the specified video. *Example: video:89917098.*

classmethod create(*channel_id*, *title*)
Creates a new collection owned by a specified channel.

Parameters

- **channel_id** (*string*) – Channel ID
- **title** (*string*) – Collection title

classmethod update(*collection_id*, *title*)
Updates the title of a specified collection.

Parameters

- **collection_id** (*string*) – Collection ID
- **title** (*string*) – Collection title

classmethod create_thumbnail(*collection_id*, *item_id*)
Adds the thumbnail of a specified collection item as the thumbnail for the specified collection.

Parameters

- **collection_id** (*string*) – Collection ID
- **item_id** (*string*) – Item ID

classmethod delete(*collection_id*)
Deletes a specified collection.

Parameters **collection_id** (*string*) – Collection ID

classmethod add_item(*collection_id*, *item_id*, *item_type*)
Adds a specified item to a specified collection.

Parameters

- **collection_id** (*string*) – Collection ID
- **item_id** (*string*) – Item ID
- **item_type** (*string*) – Type of the item. Example: *video*.

classmethod delete_item(*collection_id*, *collection_item_id*)
Deletes a specified collection item from a specified collection.

Parameters

- **collection_id** (*string*) – Collection ID
- **collection_item_id** (*string*) – Collection Item ID

classmethod move_item(*collection_id*, *collection_item_id*, *position*)
Deletes a specified collection item from a specified collection.

Parameters

- **collection_id** (*string*) – Collection ID
- **collection_item_id** (*string*) – Collection Item ID
- **position** (*int*) – New item position

Communities

class twitch.api.communities.**Communities**
This class provides methods for easy access to Twitch Communities API.

classmethod get_by_name(*community_name*)
Gets a specified community.

Parameters **community_name** (*string*) – Name of the community

classmethod get_by_id(*community_id*)
Gets a Community object based on specified user id.

Parameters **community_id** (*string*) – Community ID

classmethod create(*name*, *summary*, *description*, *rules*)
Creates a community.

Parameters

- **name** (*string*) – Community name.
- **summary** (*string*) – Short description of the community.
- **description** (*string*) – Long description of the community.
- **rules** (*string*) – Rules displayed when viewing a community page.

classmethod update(*community_id*, *summary*, *description*, *rules*, *email*)
Updates a community.

Parameters

- **community_id** (*string*) – Community ID
- **summary** (*string*) – Short description of the community.
- **description** (*string*) – Long description of the community.
- **rules** (*string*) – Rules displayed when viewing a community page.
- **email** (*string*) – Email address of the community owner.

```
>>> from twitch import TwitchClient
>>> client = TwitchClient('<my client id>', '<my oauth token>')
>>> community = client.communities.update(12345, 'foo', 'bar')
```

Games

class twitch.api.games.**Games**
This class provides methods for easy access to Twitch Games API.

classmethod get_top(*limit*, *offset*)
Gets a list of games sorted by number of current viewers.

Parameters

- **limit** (*int*) – Maximum number of objects to return. Default 25. Maximum 100.
- **offset** (*int*) – Object offset for pagination of result. Default 0.

```
>>> from twitch import TwitchClient
>>> client = TwitchClient('<my client id>')
>>> games = client.games.get_top()
```

Ingests

class twitch.api.ingests.**Ingests**
This class provides methods for easy access to Twitch Ingests API.

classmethod get_server_list()
Gets a list of ingest servers.

```
>>> from twitch import TwitchClient
>>> client = TwitchClient('<my client id>')
>>> ingests = client.ingests.get_server_list()
```

Search

class twitch.api.search.**Search**

This class provides methods for easy access to Twitch Search API.

classmethod channels(*query*, *limit*, *offset*)

Searches for channels based on a specified query parameter.

Parameters

- **query** (*string*) – Search query
- **limit** (*int*) – Maximum number of objects to return. Default 25. Maximum 100.
- **offset** (*int*) – Object offset for pagination of result. Default 0.

```
>>> from twitch import TwitchClient
>>> client = TwitchClient('<my client id>')
>>> channels = client.search.channels('lirik', limit=69, offset=420)
```

classmethod games(*query*, *live*)

Searches for games based on a specified query parameter.

Parameters

- **query** (*string*) – Search query
- **live** (*boolean*) – If *True*, returns only games that are live on at least one channel. Default: *False*.

classmethod streams(*query*, *limit*, *offset*, *hls*)

Searches for streams based on a specified query parameter.

Parameters

- **query** (*string*) – Search query
- **limit** (*int*) – Maximum number of objects to return. Default 25. Maximum 100.
- **offset** (*int*) – Object offset for pagination of result. Default 0.
- **hls** (*boolean*) – If *True*, returns only HLS streams; if *False*, only RTMP streams; if *None*, both HLS and RTMP streams.

Streams

class twitch.api.streams.**Streams**

This class provides methods for easy access to Twitch Streams API.

classmethod get_stream_by_user(*channel_id*, *stream_type*)

Gets stream information for a specified user.

Parameters

- **channel_id** (*string*) – ID of the channel you want to get information of

- **stream_type** (*string*) – Constrains the type of streams returned. Default STREAM_TYPE_LIVE.

classmethod get_live_streams(*channel*, *game*, *language*, *stream_type*, *limit*, *offset*)
Gets a list of live streams.

Parameters

- **channel** (*string*) – Comma-separated list of channel IDs you want to get
- **game** (*string*) – Game of the streams returned
- **language** (*string*) – Constrains the language of the streams returned
- **stream_type** (*string*) – Constrains the type of streams returned. Default STREAM_TYPE_LIVE.
- **limit** (*int*) – Maximum number of objects to return. Default 25. Maximum 100.
- **offset** (*int*) – Object offset for pagination of result. Default 0.

classmethod get_summary(*game*)
Gets a list of summaries of live streams.

Parameters **game** (*string*) – Game of the streams returned

classmethod get_featured(*limit*, *offset*)
Gets a list of all featured live streams.

Parameters

- **limit** (*int*) – Maximum number of objects to return. Default 25. Maximum 100.
- **offset** (*int*) – Object offset for pagination of result. Default 0.

classmethod get_followed(*stream_type*, *limit*, *offset*)
Gets a list of online streams a user is following, based on a specified OAuth token.

Parameters

- **stream_type** (*string*) – Constrains the type of streams returned. Default STREAM_TYPE_LIVE.
- **limit** (*int*) – Maximum number of objects to return. Default 25. Maximum 100.
- **offset** (*int*) – Object offset for pagination of result. Default 0.

```
>>> from twitch import TwitchClient
>>> client = TwitchClient('<my client id>', '<my oauth token>')
>>> followed = client.streams.get_followed()
```

classmethod get_streams_in_community(*community_id*)
Gets a list of streams in a community. (From Twitch forum Communities API Release)

Parameters **community_id** (*string*) – Community ID

```
>>> from twitch import TwitchClient
>>> client = TwitchClient('<my client id>')
>>> streams = client.streams.get_streams_in_community('5181e78f-2280-42a6-
↪873d-758e25a7c313')
```

Teams

class twitch.api.teams.**Teams**
This class provides methods for easy access to Twitch Teams API.

classmethod get(*team_name*)
Gets a specified team object.

Parameters **team_name** (*string*) – Name of the team you want to get information of

classmethod get_all(*limit*, *offset*)
Gets all active teams.

Parameters

- **limit** (*int*) – Maximum number of objects to return. Default 25. Maximum 100.
- **offset** (*int*) – Object offset for pagination of result. Default 0.

```
>>> from twitch import TwitchClient
>>> client = TwitchClient('<my client id>')
>>> teams = client.teams.get_all()
```

Users

class twitch.api.users.**Users**
This class provides methods for easy access to Twitch Users API.

classmethod get()
Gets a user object based on the OAuth token provided.

classmethod get_by_id(*user_id*)
Gets a user object based on specified user id.

Parameters **user_id** (*'string*) – User ID

classmethod get_emotes(*user_id*)
Gets a list of the emojis and emoticons that the specified user can use in chat

Parameters **user_id** (*'string*) – User ID

classmethod check_subscribed_to_channel(*user_id*, *channel_id*)
Checks if a specified user is subscribed to a specified channel.

Parameters

- **user_id** (*'string*) – User ID
- **channel_id** (*'string*) – ID of the channel you want to check if user is subscribed to

classmethod get_follows(*user_id*, *limit*, *offset*, *direction*, *sort_by*)
Gets a list of all channels followed by a specified user.

Parameters

- **user_id** (*'string*) – User ID
- **limit** (*int*) – Maximum number of objects to return. Default 25. Maximum 100.
- **offset** (*int*) – Object offset for pagination of result. Default 0.
- **direction** (*string*) – Sorting direction. Default DIRECTION_DESC.
- **sort_by** (*string*) – Sorting key. Default USERS_SORT_BY_CREATED_AT.

classmethod check_follows_channel(*user_id*, *channel_id*)
Checks if a specified user follows a specified channel.

Parameters

- **user_id** (*'string*) – User ID
- **channel_id** (*'string*) – ID of the channel you want to check if user is following

classmethod follow_channel(*user_id*, *channel_id*, *notifications*)
Adds a specified user to the followers of a specified channel.

Parameters

- **user_id** (*'string*) – User ID
- **channel_id** (*'string*) – ID of the channel you want user to follow
- **notifications** (*boolean*) – If true, the user gets email or push notifications when the channel goes live. Default False.

classmethod unfollow_channel(*user_id*, *channel_id*)
Deletes a specified user from the followers of a specified channel.

Parameters

- **user_id** (*'string*) – User ID
- **channel_id** (*'string*) – ID of the channel you want user to unfollow

classmethod get_user_block_list(*user_id*, *limit*, *offset*)
Gets a user's block list.

Parameters

- **user_id** (*'string*) – User ID
- **limit** (*int*) – Maximum number of objects to return. Default 25. Maximum 100.
- **offset** (*int*) – Object offset for pagination of result. Default 0.

classmethod block_user(*user_id*, *blocked_user_id*)
Blocks a user.

Parameters

- **user_id** (*'string*) – User ID
- **blocked_user_id** (*'string*) – ID of the user you wish to block

classmethod unblock_user(*user_id*, *blocked_user_id*)
Unblocks a user.

Parameters

- **user_id** (*'string*) – User ID
- **blocked_user_id** (*'string*) – ID of the user you wish to unblock

classmethod translate_usernames_to_ids(*usernames*)
Translates a list of usernames to user ID's.

Parameters **usernames** (*list[string]*) – List of usernames you wish to get ID's of

```
>>> from twitch import TwitchClient
>>> client = TwitchClient('<my client id>')
>>> users = client.users.translate_usernames_to_ids(['lirik', 'giantwaffle'])
>>>
>>> for user in users:
>>>     print('{}: {}'.format(user.name, user.id))
'lirik: 23161357'
'giantwaffle: 22552479'
```

Videos

class twitch.api.videos.**Videos**

This class provides methods for easy access to Twitch Videos API.

classmethod get_by_id(*video_id*)

Gets a Video object based on specified video ID.

Parameters **video_id** (*'string*) – Video ID

classmethod get_top(*limit*, *offset*, *game*, *period*, *broadcast_type*)

Gets a list of top videos.

Parameters

- **limit** (*int*) – Maximum number of objects to return. Default 10. Maximum 100.
- **offset** (*int*) – Object offset for pagination of result. Default 0.
- **game** (*string*) – Name of the game.
- **period** (*string*) – Window of time to search. Default PERIOD_WEEK.
- **broadcast_type** (*string*) – Type of broadcast returned. Default BROADCAST_TYPE_HIGHLIGHT.

classmethod get_followed_videos(*limit*, *offset*, *broadcast_type*)

Gets a list of followed videos based on a specified OAuth token.

Parameters

- **limit** (*int*) – Maximum number of objects to return. Default 10. Maximum 100.
- **offset** (*int*) – Object offset for pagination of result. Default 0.
- **broadcast_type** (*string*) – Type of broadcast returned. Default BROADCAST_TYPE_HIGHLIGHT.

```
>>> from twitch import TwitchClient
>>> client = TwitchClient('<my client id>', '<my oauth token>')
>>> videos = client.videos.get_followed_videos()
```

Index

I

M

R

S

T

U

V