PyTube Documentation

Release 0.1a

Noah Silas

December 17, 2013

Contents

Contents:

CHAPTER 1

PyTube Overview

PyTube is a pure python library for accessing the YouTube Data API.

1.1 Example

Display the 50 most recent videos from the 'mahalobaking' user:

```
import pytube
client = pytube.Client('my-app-identifier')
videos = client.user_videos('mahalobaking')
for video in videos[:50]:
    sys.stdout.write('%s %s\n' % (video.id, video.title))
```

1.2 Fetching Videos

1.2.1 Getting a Specific Video

client.video(*video_id*) Gets the video with youtube id *video_id*. If the video id doesn't exist, or has been deleted, this will raise *pytube.NoSuchVideoException*. If the video is private and the client is not authenticated to a user with permission to see it, this will raise *pytube.PrivateVideoException*.

1.2.2 Getting Videos from a Channel

client.user_videos(*username='default*) Returns a video stream of videos in *username*'s channel. If the client is authenticated, you may omit the username to get the authenticated user's stream.

1.2.3 Searching for Videos

client.video_search(*q=None, **query*) Returns a VideoStream of videos matching the query parameters. You can perform a basic search very simply:

```
vids = c.video_search('search terms')
```

You can also pass any of the parameters accepted by the gdata API:

```
vids = c.video_search(
    category='cars|music',  # category is cars or music
    author='schmoyoho',     # in schmoyoho's channel
    orderby='published',    # ordered by published date
    safeSearch='strict',    # 'moderate' and 'none' are the other safe search options
)
```

1.3 Video objects

1.3.1 Attributes

Videos fetched from the API should have the following attributes available.

- id
- title
- author
- category
- category.label
- keywords
- published
- updated
- description
- duration
- aspect_ratio
- private - True if this is a private video, False otherwise
- access_control - a dictionary mapping 'actions' to 'permissions'

1.3.2 Possible Attributes

The following attributes may not be set on video objects, depending on the privacy settings on the video you have fetched.

- like_count
- dislike_count
- favorite_count
- view_count
- comment_count
- uploaded - the datetime that the video was uploaded

1.3.3 Available Streams

Depending on the youtube API result, any or all of the following streams may be available on Video instances:

Video.‘related_videos‘ A stream of videos that youtube believes are related to this video.

Video.‘video_responses‘ A stream of videos that are video responses to this video.

Video.‘comments‘ A stream of comments made on this video

1.3.4 Updating A Video

Videos owned by a user who has authenticated this client can be updated. To update a video, you simply update it’s attributes and then call Video.‘update‘(). The following attributes may be updated this way:

- title
- description
- category
- keywords
- access_control
- private

CHAPTER 2

The Pytube Client

2.1 Instantiating

When creating a Client, you must supply an application identifier. This is an arbitrary string you can choose to identify the application submitting API requests.

You may also specify a developer key. This is recommended for high volume applications.

2.2 Authenticating

Authenticating the client enables a number of actions to be taken on behalf of a user and may cause some API results for content that the authenticated user to include additional or private data.

You can authenticate a client against a youtube account through google's ClientLogin or AuthSub:

```
c = pytube.Client('appid')
# Client Login
c.authenticate(channelname, password)

# Authsub Login
c.authenticate(authsub=token)
```

2.2.1 Captcha Requests When Authenticating

Sometimes google will request that you complete a captcha when authenticating via ClientLogin. Here is an example of how to handle this in a console-based app:

```
while True:
    captcha = None
    try:
        c.authenticate(channelname, password, captcha)
        break
    except pytube.CaptchaRequired, captcha:
        print "please solve the captcha at %s" % captcha.captcha
        captcha.solved = raw_input('>>> ')
```

pytube.CaptchaRequired has the following attributes:

- *token* - a unique token that must be submitted with the solved captcha
- *captcha* - the url of a captcha image to be solved

If you don't want to store the exception, you may create an object to pass into authenticate from a token and a solved captcha. Anything supplying the attributes *token* and *solved* will work:

```
class SolvedCaptcha(object):
    def __init__(self, token, solved):
        self.token = token
        self.solved = solved
```

CHAPTER 3

PyTube Streams

Most PyTube methods that return a collection of objects actually return a *Stream*. Streams have some nice properties:

- Streams will perform the minimum number of API queries necessary to return all of the results you have requested.
- For some common behaviors (iterating, fetching the first N objects) the stream can maintain an internal cache, allowing you to iterate the stream multiple times or fetch an instance from the stream without sending additional youtube API queries.

3.1 Use Streams like lists

```
videos = client.user_videos('BeyonceVEVO')

# iterate across a stream
for video in videos:
    print video.title

# slice a stream
new_videos = videos[:10]
older_videos = videos[10:20]

# get one item from a stream
video = videos[7]
```

3.2 Streams can only retrieve 1000 results

Due to limitations in the youtube API, streams may only return the first 1000 results. This can lead to confusing behavior: when you ask a stream for its length it returns the total number of items in the collection, not the number of items that it will return.:

```
>>> len(videos)
5223
>>> len(list(videos))
1000
```

CHAPTER 4

Subscriptions

4.1 Get the channels a user is subscribed to

Client.user_subscriptions(*username*) Authenticated clients may omit the *username* parameter to fetch a stream of usernames that the authenticated user is subscribed to.:

```
c = pytube.Client('app_id')
c.user_subscriptions('TheOfficialSkrillex)
```

4.2 Subscribe the Authenticated User to a channel

Client.subscribe(*username*) Authenticated clients may call this method to subscribe to a channel.

PyTube doesn't have an unsubscribe method yet. =(