Youtube explorer

Support Document

Honey Kakkar

honeykakkar@outlook.com

Contents

[Abstract 2](#_Toc469917447)

[Getting started 2](#_Toc469917448)

[Obtaining authorization credentials 2](#_Toc469917449)

[Creating OAuth 2.0 credentials 3](#_Toc469917450)

[Creating API Keys 3](#_Toc469917451)

[Resources 3](#_Toc469917452)

[Operations 3](#_Toc469917453)

[Quota usage 4](#_Toc469917454)

[Classes contained in the project 4](#_Toc469917455)

[Auth 4](#_Toc469917456)

[Channels 4](#_Toc469917457)

[ChannelUploads 6](#_Toc469917458)

[CommentsAndReplies 7](#_Toc469917459)

# Abstract

This document is intended for developers who want to use YouTube-Explorer application that interacts with YouTube through YouTube Data API. It also provides an overview of the different functions that the application with the help of the API supports.

# Getting started

Before you start

1. You need a [Google Account](https://www.google.com/accounts/NewAccount) to access the Google Developers Console, request an API key, and register your application.
2. Create a project in the [Google Developers Console](https://console.developers.google.com/) and [obtain authorization credentials](https://developers.google.com/youtube/registering_an_application) so your application on the console can submit API requests. YouTube-Explorer uses those credentials to collect data from YouTube.
3. After creating your project, make sure the YouTube Data API is one of the services that your application is registered to use:
   1. Go to the [Developers Console](https://console.developers.google.com/) and select the project that you just registered.
   2. [Open the API Library](https://console.developers.google.com/apis/library?project=_) in the Google Developers Console. If prompted, select a project or create a new one. In the list of APIs, make sure the status is **ON** for the **YouTube Data API v3**.

# Obtaining authorization credentials

1. Your application must have authorization credentials to be able to use the YouTube Data API. Assuming that you have created a project on Google Developers Console, Open the [Credentials page](https://console.developers.google.com/apis/credentials?project=_).
2. The API supports API keys and OAuth 2.0 credentials. In YouTube-Explorer, I have implemented both methods for the authorization of the requests to help the readers to choose as per their convenience.

## Creating OAuth 2.0 credentials

You can generate OAuth 2.0 credentials for web applications, service accounts, or installed applications.

1. On the credentials page of the project, click **Create credentials > OAuth Client ID**
2. Select the appropriate type of the project to **create the client ID and client secret**
3. Copy both fields to file named “**client\_secrets.json**” under “/src/main/**resources**” directory of the project.

The API reduces the amount of code you need to handle OAuth 2.0. **Auth** class in the project contains methods for authorizing a user and caching credentials.

## Creating API Keys

1. Create an API key by clicking **Create credentials > API key** and selecting the appropriate key type. Then enter the additional data for that key type.
2. Once you obtain the API key, copy and paste the key to the file named “**youtube.properties**” under “/src/main/**resources**” directory of the project.

After following the above mentioned steps to obtain the credentials, the authorization process is complete to use YouTube-Explorer.

# Resources

A resource is an individual data entity with a unique identifier.  YouTube-Explorer (as of December 17, 2016) uses following resources to interact with YouTube:

* **channel** - Contains information about a single YouTube channel
* **playlist** - Represents a single YouTube playlist. A playlist is a collection of videos that can be viewed sequentially and shared with other users.
* **playlistItem** - Identifies a resource, such as a video, that is part of a playlist. The playlistItem resource also contains details that explain how the included resource is used in the playlist.
* **video** - Represents a single YouTube video.

# Operations

Following are the most common methods that the API supports. Some resources also support other methods that perform functions more specific to those resources.

* **List** - Retrieves (GET) a list of zero or more resources.
* **Insert** - Creates (POST) a new resource.
* **Update** - Modifies (PUT) an existing resource to reflect data in your request.
* **Delete** - Removes (DELETE) a specific resource.

YouTube-Explorer as of December 17, 2016 contains classes which use **list** method to retrieve the data from YouTube.

# Quota usage

The YouTube Data API uses a quota to ensure that developers use the service as intended. Different types of operations have different quota costs. A simple read operation that only retrieves the ID of each returned resource has a cost of approximately 1 unit.

An API request that returns resource data must specify the resource parts that the request retrieves. Each part then adds approximately 2 units to the request's quota cost. As such, a videos.list request that only retrieves the snippet part for each video might have a cost of 3 units.

# Classes contained in the project

## Auth

It is a shared class used by every class of the project. It contains methods for authorizing a user and caching credentials.

## Channels

The list method of channels resource of the API returns a collection of zero or more **channel** resources that match the request criteria. In class “Channels” of YouTube Explorer project, I am retrieving the object of channel based on its channel id.

One may like to visit [Channels resource](https://developers.google.com/youtube/v3/docs/channels) to check the fields associated with a channel object. In order to perform any operation on a channel, one must have a channel object in order to send the request to API.

This class contains following methods:

* **importChannelLinks** – Method to fetch the channel links from an excel sheet in the current directory.
* **getChannel** - Method to collect the Channel object, with its content details and the statistics, of a given channel ID.

The ***content details*** of a channel includes:

"relatedPlaylists": {

"likes"

"favorites"

"uploads"

"watchHistory"

"watchLater"

}

***Statistics*** include:

"viewCount"

"commentCount"

"subscriberCount"

"hiddenSubscriberCount"

"videoCount"

* **getChannelUploadList** - Method to get the playlist of videos uploaded by the given channel
* **getUploadedVideos** - Method to get the list of videos (video IDs) uploaded by the channel
* **getVideoDetails** - Method to collect the statistical information of a video. Statistical information includes
  + Likes
  + Dislikes
  + View Count
  + Comment Count
* **getVideoContents -** Method to collect the content-details object associated with a video ID. ContentDetails of a video includes:
  + "duration"
  + "dimension"
  + "definition"
  + "caption"
* **getVideoSnippet** - Method to collect the snippet object associated with a video ID. Snippet of a video includes:
  + "publishedAt"
  + "channelId"
  + "title"
  + "description"
  + "channelTitle"
  + "categoryId"
* **establishConnection -** Method to establish a connection to a MySQL database
* **exportChannelDataToDB** - Method to push the information collected of channels to the database
* **exportVideosDataToDB** - Method to push the information collected of uploaded videos of channels to the database

## ChannelUploads

Class contains methods to collect the comments and their respective replies of the videos uploaded by a channel.

This class contains following methods:

* **getVideoDetails** - Method to collect the statistical information of a given video. Statistical information includes
  + Likes
  + Dislikes
  + View Count
  + Comment Count
* **collectAllComments** - Method to collect the maximum number of top level comments of a given YouTube video ID. This method collects the comments in the order of the "relevance"
* **collectAllReplies** - Method to collect all replies from each top level comment from the list of comments
* **establishConnection** - Method to establish a connection to a MySQL database
* **pushComments** - Method to insert the top level comments of the video into a "Comments" table in the database.
* **pushReplies** - Method to insert the replies of the comments of the video into a "Replies" table in the database.
* **exportToMySQLDB** - Method to export all the collected comments and the replies to a MySQL database
* **getVideosIdsFromDB** - Method to collect the video IDs of uploaded videos by the channel from the database

## CommentsAndReplies

This class contains methods to collect the comments and their respective replies based on a given YouTube video ID. It contains all the methods contained in ChannelUploads serving the same purpose to collect and push the comments and their replies to the database.

For a given video, first all top-level comments are collected. To collect the respective replies on each comment, the list of top-level comments is passed to method “collectAllReplies” in the class to have a list of all replies on the video.