**Chapter 4 Examples Part-1**

The examples in this document focus on retrieving data from Youtube by using the Youtube’s official API.

**Login:** This example demonstrates how to create an app through the Google Developers Console and connect a python script to that app. All other scripts in this chapter re-use this login script to reduce the amount of coding on other pages.

Original Tutorial: <https://www.thepythoncode.com/article/using-youtube-api-in-python>

Google Developers Console:

<https://console.developers.google.com>

Information on Quota Cost: <https://developers.google.com/youtube/v3/determine_quota_cost>

Information on Scopes when setting up an app:

<https://developers.google.com/identity/protocols/oauth2/scopes>

Required Python Library:

pip3 install --upgrade google-api-python-client google-auth-httplib2 google-auth-oauthlib

# Useful Solution Link for [Error 403: access\_denied The developer hasn’t given you access to this app despite a new project being created](https://stackoverflow.com/questions/65756266/error-403-access-denied-the-developer-hasn-t-given-you-access-to-this-app-despi)

<https://stackoverflow.com/questions/65756266/error-403-access-denied-the-developer-hasn-t-given-you-access-to-this-app-despi>

login.py

from googleapiclient.discovery import build

from google\_auth\_oauthlib.flow import InstalledAppFlow

from google.auth.transport.requests import Request

import os

import pickle

#This is where we declare the scope of all the youtube analytics based pages we want to build.

#This scope indicates that all the apps that will login to Yotube's Servers will only read the youtube data and will not change anything

SCOPES = ["https://www.googleapis.com/auth/youtube.readonly"]

#The code below logs into the Youtube's Servers by using the credentials.json library

os.environ["OAUTHLIB\_INSECURE\_TRANSPORT"] = "1"

api\_service\_name = "youtube"

api\_version = "v3"

#Make sure the json file name below matches your json file which you put into your project folder

client\_secrets\_file = "credentials.json"

creds = None

# the file token.pickle stores the user's access and refresh tokens, and is

# created automatically when the authorization flow completes for the first time

if os.path.exists("token.pickle"):

with open("token.pickle", "rb") as token:

creds = pickle.load(token)

# if there are no (valid) credentials availablle, let the user log in.

if not creds or not creds.valid:

if creds and creds.expired and creds.refresh\_token:

creds.refresh(Request())

else:

flow = InstalledAppFlow.from\_client\_secrets\_file(client\_secrets\_file, SCOPES)

creds = flow.run\_local\_server(port=0)

# save the credentials for the next run

with open("token.pickle", "wb") as token:

pickle.dump(creds, token)

youtube = build(api\_service\_name, api\_version, credentials=creds)

**Important Note:** For security oriented purposes, the Tokens expire! When they do, the python scripts are unable to communicate with the Youtube’s servers. When your token expires, delete the token file from the project. Next, re-run the login.py script to create a new token.

**Example 1:** This example demonstrates how to connect a script to the YouTube Analytics API via the previous login file and then retrieve channel statistics from Youtube.

This page shows the data we can retrieve from a YouTube Channel

<https://developers.google.com/youtube/v3/docs/channels?hl=en>

Google is going to build the source code for us. Click on the </> icon next to the list by Youtube forUserName. This try method does not consume daily quota.

<https://developers.google.com/youtube/v3/docs/channels/list?hl=en&apix_params=%7B%22part%22%3A%5B%22snippet%2CcontentDetails%2Cstatistics%22%5D%2C%22forUsername%22%3A%22JayzTwoCents%20%22%7D>

I am using the method described here to save Youtube query results into a dataframe

<https://stackoverflow.com/questions/69790171/select-specific-keys-from-youtube-api-response>

This code is based on username and it does not always work! This is how we can retrieve Youtube Channel ID, which has higher chance of success.I like grabbing it from the page source. Search for <https://www.youtube.com/channel/> in the page source code.

<https://mixedanalytics.com/blog/find-a-youtube-channel-id/#pagesource>

import login

import pandas

request = login.youtube.channels().list(

part="snippet,contentDetails,statistics",

forUsername="JayzTwoCents"

)

response = request.execute()

#print(response)

#Note the api does not display total number of likes and uploads per channel

#Response output is difficult to work with. I select outputs I like from the response and put them into a dataframe

results\_dataframe= pandas.json\_normalize(response['items'])[['snippet.title','snippet.description','snippet.country','statistics.viewCount','statistics.subscriberCount','statistics.videoCount']]

results\_dataframe.to\_csv("example1results.csv", index=False)

This code is based on channel ID and it has higher success rate:

Google is going to build the source code for us. Click on the </> icon next to the list by id. This try method does not consume daily quota.

**Example 1b:**

import login

import pandas

request = login.youtube.channels().list(

part="snippet,contentDetails,statistics",

id="UCeY0bbntWzzVIaj2z3QigXg"

)

response = request.execute()

#print(response)

#Note the api does not display total number of likes and uploads per channel

#Response output is difficult to work with. I select outputs I like from the response and put them into a dataframe

results\_dataframe= pandas.json\_normalize(response['items'])[['snippet.title','snippet.description','statistics.viewCount','statistics.subscriberCount','statistics.videoCount']]

results\_dataframe.to\_csv("example1bresults.csv", index=False)

**Example 2:** This example demonstrates how to retrieve multiple Youtube Content Creators’ channel information. The user of the script decides how many loops to make. The loop logic is from here: <https://tutorial.eyehunts.com/python/while-loop-yes-or-no-python-example-code/>

import login

import pandas

appended\_data\_list = []

while True:

userinput = input("Enter Youtube Channel ID: ")

# your code

request = login.youtube.channels().list(

part="snippet,contentDetails,statistics",

id=userinput

)

response = request.execute()

# print(response)

# Note the api does not display total number of likes and uploads per channel

# Response output is difficult to work with. I select outputs I like from the response and put them into a dataframe

results\_dataframe = pandas.json\_normalize(response['items'])[

['snippet.title', 'snippet.description', 'snippet.country', 'statistics.viewCount',

'statistics.subscriberCount', 'statistics.videoCount']]

appended\_data\_list.append(results\_dataframe)

cont = input("Another one? yes/no > ")

while cont.lower() not in ("yes", "no"):

cont = input("Another one? yes/no > ")

if cont == "no":

print("Break")

appended\_data\_dataframe = pandas.concat(appended\_data\_list)

appended\_data\_dataframe.to\_csv("example2results.csv", index=False)

break

**Example 3:** This example demonstrates how to retrieve details of a Youtube video and save those details to a csv file.

This page shows how to retrieve the details of a Youtube video:

<https://developers.google.com/youtube/v3/docs/videos/list>

How to find Youtube Video ID: <https://www.easterseals.com/assets/includes/youtube-id.html#:~:text=How%20to%20get%20a%20YouTube,after%20the%20v%3D%20URL%20parameter>.

Example Youtube Video ID: tNxUxm3-658

import login

import pandas

request = login.youtube.videos().list(

part="snippet,contentDetails,statistics",

id=" X3byz3txpso"

)

response = request.execute()

#print(response)

results\_dataframe= pandas.json\_normalize(response['items'])[['snippet.channelTitle','snippet.title','statistics.viewCount','statistics.likeCount','statistics.commentCount', 'statistics.favoriteCount']]

results\_dataframe.to\_csv("example3results.csv", index=False)

**Example 4:** This example demonstrates how to retrieve the details of multiple Youtube videos, rename default column headings in the dataframe, and save the results to a csv file.

This page shows how to retrieve the details of multiple Youtube videos via their IDs:

<https://developers.google.com/youtube/v3/docs/videos/list>

Choose the **list (multiple video IDs)** option above.

Example Youtube Video IDs: v7ICKbmysA (Does not work), EXh6oAJH2Zg (works), 6eH2BItdo0M (works)

**FYI: Not all video IDs are scrapeable. Try Youtube’s API before the code below to see if you can scrape data from a video without consuming your daily quote limit.**

import login

import pandas

request = login.youtube.videos().list(

part="snippet,contentDetails,statistics",

id="bISWIk5pUH0,tWbjy\_sCbUA,p79H\_XOwpZo,CmNMfRpjknA"

)

response = request.execute()

#print(response)

results\_dataframe= pandas.json\_normalize(response['items'])[['snippet.channelTitle','snippet.title','statistics.viewCount','statistics.likeCount','statistics.commentCount', 'statistics.favoriteCount']]

#The code below renames the column names in the dataframe

results\_dataframe.rename(columns= {'snippet.channelTitle': 'Channel\_Title','snippet.title': 'Video\_Title', 'statistics.viewCount': 'View\_Count', 'statistics.likeCount':'Like\_Count','statistics.commentCount':'Comment\_Count','statistics.favoriteCount':'Favorite\_Count'}, inplace=True)

results\_dataframe.to\_csv("example4results.csv", index=False)

**Example 5:** This example demonstrates how to retrieve a maximum of 100 comments and replies from a Youtube Video.

We are going to use the following tutorial for this example:

<https://developers.google.com/youtube/v3/docs/commentThreads/list>

Select list (by video ID) approach from the list. Enter VideoID and MaxResults parameters. Sort by relevance to match the results on Youtube

FYI: Google OAuth 2.0 authentication does not work! Instead use the API key approach

In order to access to different parts of the JSON file, I am using the following tutorial:

<https://www.thepythoncode.com/article/using-youtube-api-in-python>

This is another good tutorial:

<https://towardsdatascience.com/how-to-build-your-own-dataset-of-youtube-comments-39a1e57aade>

See their extracting comments from the Youtube Videos Section.

#API Method to do Example 5 with the replies

# -\*- coding: utf-8 -\*-

# Sample Python code for youtube.commentThreads.list

# See instructions for running these code samples locally:

# https://developers.google.com/explorer-help/code-samples#python

import os

import pandas

import googleapiclient.discovery

def main():

# Disable OAuthlib's HTTPS verification when running locally.

# \*DO NOT\* leave this option enabled in production.

os.environ["OAUTHLIB\_INSECURE\_TRANSPORT"] = "1"

api\_service\_name = "youtube"

api\_version = "v3"

DEVELOPER\_KEY = "YOUR API KEY"

youtube = googleapiclient.discovery.build(

api\_service\_name, api\_version, developerKey = DEVELOPER\_KEY)

request = youtube.commentThreads().list(

part="snippet,replies",

maxResults=100,

videoId="tNxUxm3-658"

)

response = request.execute()

print(response)

# I am creating an empty dictionary here for the results I like to save to a CSV file

output = {"CommentID": [], "UserID": [], "Comment": [], "Number\_of\_Likes": [], "Number\_of\_Replies": [],

"Updated\_At": []}

for item in response['items']:

# I am accessing different parts of the JSON response file that I consider important

comment\_id = item['snippet']['topLevelComment']['id']

userID = item["snippet"]["topLevelComment"]["snippet"]["authorDisplayName"]

comment = item['snippet']['topLevelComment']['snippet']['textDisplay']

like\_count = item['snippet']['topLevelComment']['snippet']['likeCount']

reply\_count = item['snippet']['totalReplyCount']

updated\_at = item["snippet"]["topLevelComment"]["snippet"]["updatedAt"]

# I am appending the results for each comment

output['CommentID'].append(comment\_id)

output['UserID'].append(userID)

output['Comment'].append(comment)

output['Number\_of\_Likes'].append(like\_count)

output['Number\_of\_Replies'].append(reply\_count)

output['Updated\_At'].append(updated\_at)

results = pandas.DataFrame.from\_dict(output, orient='index')

results = results.transpose()

results.to\_csv('example5\_results.csv', index=False)

if \_\_name\_\_ == "\_\_main\_\_":

main()

**Example 6:** This example demonstrates how to download the captions of a Youtube Video and save the results to a csv file.

Requirement:

pip3 install youtube-transcript-api

from youtube\_transcript\_api import YouTubeTranscriptApi

import pandas

# assigning srt variable with the list

# of dictonaries obtained by the get\_transcript() function

caption\_list = YouTubeTranscriptApi.get\_transcript("6eH2BItdo0M")

#print(caption\_list)

caption\_dataframe = pandas.DataFrame(caption\_list)

caption\_dataframe.to\_csv("example6results.csv", index=False)

**Example 7:** This example demonstrates how to convert a Youtube video’s captions into string format future analysis purposes.

import pandas

#I am creating a dictionary here titled inputdata

inputdata={}

inputdata = pandas.read\_csv('example6results.csv').to\_dict()

# I created a new dictionary here for the text column in my csv file

caption\_text\_dictionary = inputdata.get('text')

# I am converting the caption dictionary to a list so I can analyze the data

caption\_text\_list = list(caption\_text\_dictionary.values())

#convert list to string

caption\_text\_instring = ''

for eachletter in caption\_text\_list:

caption\_text\_instring += eachletter

print(caption\_text\_instring)