****

**Scrapy Automation**

**Billboard Top Artists**

**Soheil Raeisi**

**Date:** 2022/04/26

Contents

[**Web scraping** 2](#_Toc101907172)

[**Project Content** 2](#_Toc101907173)

[**Step by Step Instruction** 3](#_Toc101907174)

[**Google Form & Data Summary** 4](#_Toc101907175)

# 

# **Web scraping**

In today’s competitive world everybody is looking for a way to innovate and make use of new technologies. Web scraping (also called web data extraction or [data scraping](https://www.zyte.com/data-extraction/)) provides a solution for those who want to get access to structured web data in an automated fashion. Web scraping is useful if the public website you want to get data from doesn’t have an API, or it does but provides only limited access to the data.

Billboard scraper is a program that firstly extracts the 5 previous week’s top artists from the Billboard website. Afterwards, the repetitive names are removed from the artists list and then be uploaded to the google drive as a google sheet file. On the google aps scripts platform two functions are provided to firstly create a form for the artists to be ranked by the respondents, and subsequently, produce an average rate according to the responses for each artist. The Whole program is attached within a folder named Billboard and you can find instruction on the way it must be used and automated in the upcoming paragraphs.

* **This program can only be utilized for Billboard website**

# **Project Content**

The project file contains different files which are explained in upcoming paragraphs.

* Billboard\_scrapy.py

This is the main program that by the aid of Scrapy framework crawls through the website and extracts the names and ranks of the artists of past 4 weeks which is called the Spider. As can be seen the URL of the web page is given to the program and by finding the date of the current webpage and deduction of weeks from it the URL of the past 4 weeks are produced, so the spider can go through 4 previous pages and according to the CSS selectors extracts the artists names and ranks.

* Data\_Cleaner.py

Data cleaner file contains the function that removes the duplicated names and all ranks. It also sorts all names alphabetically and save them to a new file named Ultimate.csv. This function uses Pandas library and it only works with CSV files.

* UPLOAD.py

This program has the responsibility for connection between the local server and the desired Google Drive. It gets the name and the type of the file which wants to be sent the google drive and also the type of the file which is intended to be saved in the drive. Undoubtedly, the email of the destination drive must be added to the code.

* GoogleAppScript.gs

Contains two functions for creation of form and data summary which can be used in the google apps script editor.

* Data.csv / Ultimate.csv

Data.csv is the raw extracted names and ranks of the top 5 artists of 4 previous weeks and Ultimate.csv is the file which contains cleaned data and is going to be uploaded to the google drive. These two files have been added to the project as examples, However it is highly recommended that do not change the names of these files while running the program, otherwise some modifications inside the codes must be done.

# **Step by Step Instruction**

Firstly copy the billboard folder on your system, then in order to use the program follow the steps:

1. Firstly, if you do not have a Google Cloud Project, it is needed to be created to be able to use the Google API:
   * Open <https://cloud.google.com> then 🡺 select ***Console*** 🡺select ***project*** 🡺***New Project***🡺 ***create***
   * Enable the API at Console 🡺***API &services***🡺***Library***🡺search “Drive API”🡺***API enabled.***
2. A very important step is that a Google service account must be created. Google API does not allow you to access the cloud with a normal account. Therefore a service account must be created. Please follow the steps:

* click on <https://cloud.google.com/docs/authentication/getting-started> then🡺click on (**Go to Create service account**)🡺select your project🡺write a name and serve account for description then ***create and continue***🡺select the ***owner*** as the ***role*** then press ***continue***🡺***Done 🡺*** now the service account is created and you can see the email address
* Now you have to create a service account key🡺click on email address🡺click on tab ***keys*** 🡺***add key***🡺***create new key***🡺choose JSON🡺by pressing ***create*** your credential JSON file which contains the service account key is downloaded. This is the file that is required to connect to Google API. Preferably put it beside the UPLOAD.py file.
* At the end you must create an environment variable to link to the credential file. Its name is **GOOGLE\_APPLICATION\_CREDENTIALS.**  🡺 just write in the command line or your IDE terminal :

In Linux or macOS: **export** **GOOGLE\_APPLICATION\_CREDENTIALS={*KEY\_PATH*}**

In Windows : **$env:GOOGLE\_APPLICATION\_CREDENTIALS={*KEY\_PATH*}**

1. Since we are going to use Google Python API to upload the file to Google drive firstly install google\_api\_python\_client library: **pip install google\_api\_python\_client**
2. The only item that needs to be changed prior to run the program is the email address which you can find it in the line 41. Currently as an example my email address is written but it has to be the address of the location where the user intends to save the data as a google sheet.
3. **Automation:** the processes of data extraction, data transformation and data loading must be scheduled to run once a month. As normally this program is run on the server which uses Linux, scheduling the program is explained for Crontab tool. However there is a subsystem for Crontab in windows called WSL which is free to download.
   * **Crontab:** there are 3 files that must be scheduled, Billboard\_scrapy.py ,Data\_Cleaner.py and UPLOAD.py. firstly it is better to change the editor in Linux to a friendly one by the command🡺export EDITOR=gedit
   * Now run the Crontab tool 🡺 crontab **–**e
   * Write these three commands to schedule 3 files to be run once a month:

0 0 1 \* \* cd {path}/billboard && scrapy crawl Billboard -O Data.csv

2 0 1 \* \* cd {path}/billboard/Data\_Cleaner.py

3 0 1 \* \* cd {path}/billboard/UPLOAD.py

* **For running the scrapy do not forget to use capital “O” to override the previous file, as the previous data is no longer needed.**

# **Google Form & Data Summary**

At this point, the ultimate file has been uploaded to the Google Drive and it can been accessed in the sheets tab. Open the uploaded sheet and on the ribbon click on the **Extensions** then **apps script.** Now the google scripts editor for the uploaded sheet is opened. From the GS file in the project copy the whole two functions and paste them in the editor. As can be seen there is a drop down tool which can be used to select the name of the desired function. Firstly choose the createFrom function, run it. The URL of the created form appears below the editor page and with that the form can be used.

To use the data summary function firstly the ID of the form must be written in the code like the example. The ID is accessible from the URL of the form. After running the dataSummary code you can see that all the average rankings are added in front of the names of the artists in the sheet.