YouTube-Subscribed-Channels-GUI

Python tool - YouTube Subscribed Channels GUI, lists user's subscribed channels with subscription dates & duration. Search, navigate results. Google API for authentication. Setup, troubleshoot, verify per Google's guide

This app is a GUI-based tool that displays a list of subscribed channels on YouTube, along with the date they were subscribed and the time since they were subscribed. The user can search for specific channels using a search bar, and the results will be displayed in a table. The user can navigate through the search results using previous and next buttons. The app uses the Google API libraries to authenticate the user and access their subscribed channels.

To use this app, follow these steps:

Get the credentials.json file from Google Cloud Platform, follow these steps:

1. Go to the Google Cloud Console (https://console.cloud.google.com/).
2. Select Your Project. If you haven't created a project yet, create one. If you have a project, select it from the dashboard.
3. Navigate to APIs & Services > Credentials.
4. Click Create Credentials > OAuth client ID.
5. Select Desktop App and give it a name.
6. Click Create.
7. Click Download to download the credentials.json file.
8. Move the credentials.json file to the same directory as your script.

Note: The verification process can be specific to your application, so it's recommended to refer to Google's official documentation for detailed and up-to-date instructions. Additionally, you might want to consult Google's support resources or forums for assistance if you encounter specific issues during the verification process.

Now the steps themselves

1. Update the Google API libraries to the latest version by running the following command in the terminal:

```
pip install --upgrade google-api-python-client google-auth google-auth-httplib2
google-auth-oauthlib tk
```

2. Create a virtual environment (replace venv with your preferred environment name) by running the following command in the terminal:

```
python -m venv venv
```

3. Activate the virtual environment:

- On Windows:

```
venv\Scripts\activate
```

- On macOS and Linux:

```
source venv/bin/activate
```

4. Install the required libraries within the virtual environment by running the following command in the terminal:

```
pip install google-api-python-client google-auth google-auth-httplib2 google-auth-oauthlib
```

5. Run the script within the activated virtual environment by running the following command in the terminal:

```
python <path_to_script>
```

Note: Replace `<path_to_script>` with the path to the script file.

To authenticate the user and access their subscribed channels, the app uses the Google API libraries. To set up the authentication, follow these steps:

1. Go to the Google Cloud Console (https://console.cloud.google.com/).

2. Select Your Project. If you haven't created a project yet, create one. If you have a project, select it from the dashboard.

3. Navigate to OAuth Consent Screen. In the left sidebar, navigate to "APIs & Services" > "OAuth consent screen."

4. Configure Consent Screen. Fill out the required information for the OAuth consent screen, including the "App name," "User support email," and other details. Make sure you provide clear and accurate information.

5. Add Scopes. Under "Scopes for Google APIs," make sure you've added the necessary scopes required for your application. In your case, you're using the YouTube Data API, so you should include the scope https://www.googleapis.com/auth/youtube.readonly.

6. Test Users. You can add email addresses of users who will be testing your application during the verification process. This will allow them to access the application even before the verification is completed.

7. Submit for Verification. Once you've configured the consent screen, you can submit it for verification. Click the "Submit for verification" button at the bottom of the page.

8. Wait for Verification. Google's verification process may take some time. You will receive emails from Google regarding the status of the verification. Be sure to respond to any additional requests for information from Google.

9. Verification Completed. Once your consent screen is verified, you should receive an email confirmation. After that, your application should be able to access Google services without the "Access blocked" error.

Note: The verification process can be specific to your application, so it's recommended to refer to Google's official documentation for detailed and up-to-date instructions. Additionally, you might want to consult Google's support resources or forums for assistance if you encounter specific issues during the verification process.

Credits:

- Website - http://beangreen247.xyz/
- LinkTree - https://linktr.ee/BeanGreen247
- Twitter - https://twitter.com/beangreen247
- Steam - https://steamcommunity.com/id/BeanGreen247/
- Github - https://github.com/BeanGreen247
- LinkedIn - https://www.linkedin.com/in/tom%C3%A1%C5%A1-mozd%C5%99e%C5%88-3382b71a6/