HLS Video Downloader Script

This script enables you to download multiple HLS (HTTP Live Streaming) videos by processing .m3u8 playlist URLs. The script reads the URLs from a file, downloads the necessary video segments, decrypts them if needed, and merges them into a single MP4 file using FFmpeg.

Prerequisites

1. Python 3.x

- The script is written in Python. You need Python version 3.x installed on your computer.
- **Installation**: Download Python from python.org and follow the installation instructions.

To check if Python is installed, open a command prompt or terminal and type:

python --version

2. Required Python Libraries

- The script requires the requests library to handle HTTP requests (for downloading the .m3u8 playlist and video segments).

To install the required libraries, use the following command:

pip install requests

To check if requests is installed, you can use:

pip show requests

3. FFmpeg

- **FFmpeg** is a powerful multimedia framework used by the script to merge video segments into one MP4 file.
- **Installation**:
    1. Download FFmpeg from the official FFmpeg website.

2. Extract the downloaded files and ensure the ffmpeg.exe file is available in your system's PATH. You can verify this by typing ffmpeg in the command prompt or terminal. If it's correctly installed, you should see FFmpeg's version information.

For Windows:
After downloading FFmpeg, follow these steps to add it to your PATH:

- Right-click **This PC** or **My Computer**, then choose **Properties**.
- Click **Advanced system settings**.
- In the System Properties window, click the **Environment Variables** button.
- Under **System Variables**, scroll down and find **Path**, select it, and click **Edit**.
- In the **Edit Environment Variable** window, click **New**, and then paste the path to the bin folder of FFmpeg (e.g., C:\ffmpeg\bin).
- Click **OK** to apply changes.

After that, run ffmpeg from the command line to verify installation. Or check the version in CMD to ensure that it has successfully installed:

ffmpeg -version

File Setup

1. m3u8_urls.txt

- This file will contain a list of .m3u8 URLs, each on a separate line. These URLs will be processed by the script to download the corresponding videos.

Example of **m3u8_urls.txt** in the file:

https://example.com/video1.m3u8?auth_key=abc123

https://example.com/video2.m3u8?auth_key=xyz456

https://example.com/video3.m3u8?auth_key=def789

2. Script

- The script will process the URLs from m3u8_urls.txt, download the necessary video segments, handle decryption (if applicable), and merge them into a single MP4 file using FFmpeg.

The script does the following:

- Downloads the playlist from each .m3u8 URL.
- Processes any key (if provided in the playlist) and downloads it.
- Uses FFmpeg to merge the video segments into an MP4 file.

How to Use the Script

1. Prepare the m3u8_urls.txt File

- Create a file called m3u8_urls.txt in the same directory as the script.
- Inside m3u8_urls.txt, list the .m3u8 URLs, each on a new line.

Example:

https://example.com/video1.m3u8?auth_key=abc123

https://example.com/video2.m3u8?auth_key=xyz456

https://example.com/video3.m3u8?auth_key=def789

2. Run the Script

- Open a command prompt or terminal in the directory where the script is located.
- Run the script using Python:

python your_script.py

3. Script Output

- The script will create a folder for each .m3u8 URL, named video_1, video_2, etc.
- Inside each folder, you will find:
  - A downloaded **key.key** file (if a decryption key is needed).

    - A playlist file named playlist.m3u8.
    - The final merged MP4 file saved as output.mp4.

4. Result

- After the script finishes, you will have individual MP4 video files named output.mp4 inside each corresponding folder (e.g., video_1/output.mp4).

Script Breakdown

1. Reading URLs

- The script reads all .m3u8 URLs from m3u8_urls.txt.

2. Downloading Key Files (if applicable)

- If the .m3u8 playlist includes an encrypted video stream (indicated by the #EXT-X-KEY tag), the script downloads the decryption key (key.key) and saves it in the same folder as the video segments.

3. Downloading Video Segments

- The script processes the playlist and downloads the video segments (.ts files) specified in the playlist.

4. Merging Segments into MP4

- Using FFmpeg, the script combines the .ts video segments into one MP4 file, ensuring it handles any decryption automatically using the key.key file if required.

Troubleshooting

1. If FFmpeg is not recognized:

- Ensure that you have installed FFmpeg and added it to your system's PATH.
- You can check if FFmpeg is correctly installed by running ffmpeg in the command prompt or terminal.

2. If requests is not installed:

- Ensure you have installed the required library by running:

```
pip install requests
```

3. If Video Download Fails:

- Check if the .m3u8 URL is still valid.
- Ensure the playlist and video segments are accessible (sometimes, video hosting services might block requests).

Additional Notes

- **Custom Output Folder**:
  You can change the output folder name by modifying the script.
- **Handling Multiple URLs**:
  The script automatically handles multiple .m3u8 URLs from m3u8_urls.txt, processing each one individually.

How You Get HLS Link?

To get the **HLS .m3u8 URL** (or full playlist), follow one of these methods using your browser:

🔍 Method 1: Inspect Network Tab (Browser Developer Tools)

1. **Open the website** where the video is playing.
2. **Right-click** → **Inspect** (or press F12).
3. Go to the **Network** tab.
4. In the **Filter box**, type:

m3u8

5. **Reload the page** or click **Play** on the video.
6. Look for a request ending in .m3u8 (e.g., index.m3u8 or playlist.m3u8).
7. **Right-click** > **Copy** > **Copy Link Address**.

👉 That copied URL is what you should paste into the M3U8_URL variable in the script.

🎥 Bonus Tip: If You Only See .ts Files

If you only see .ts segment files and **no .m3u8**, do this:

- Search the **page source** (Ctrl+U or Cmd+U) for .m3u8
- Or search the **JavaScript files** being loaded (.js files in the Network tab) – some dynamically load the playlist URL