

파이썬을 이용한 유튜브 다운로더 만들기

1. yt-dlp 라이브러리
2. 유튜브 동영상을 다운로드 하는 기본 소스 코드
3. ydl_opts 딕셔너리
4. 4. ffmpeg 설치 및 환경변수 등록
5. 도전 과제

1. yt-dlp

- YouTube와 같은 사이트에서 동영상과 오디오를 다운로드하는데 사용되는 Python 라이브러리.
- youtube-dl을 기반으로 만들어짐.

항목	Youtube-dl	Yt-dlp
업데이트 속도	상대적으로 느림	상대적으로 빠름
추가 기능	기본적인 다운로드 기능 제공	다운로드 속도 제한, 쿠키 처리, 플레이리스트 처리 등 추가 기능 제공
사이트 지원	수백 개 사이트 지원	Youtube-dl이 지원하지 않는 일부 사이트 포함, 더 많은 사이트 지원
개발 활동	2021년 9월 기준으로 잠정 중단	2021년 9월 기준으로 활발하게 진행 중

1. yt-dlp

- 주요기능

1. **다양한 사이트 지원** : YouTube 외에도 Vimeo, SoundCloud, Twitch 등 수백 개 이상의 사이트에서 미디어를 다운로드할 수 있음.
2. **포맷 선택** : 다운로드할 미디어의 포맷을 선택할 수 있음. 예) 동영상을 MP4, WebM, FLV 등의 포맷으로, 오디오를 MP3, Ogg, Vorbis, M4A 등의 포맷으로 다운로드할 수 있음.
3. **품질 선택** : 미디어의 품질을 선택. 예) 동영상의 해상도를 선택하거나, 오디오의 Bit rate 를 선택할 수 있음.

1. yt-dlp

- 주요기능

4. **자막 다운로드** : 동영상의 자막을 다운로드할 수 있음. 사용 가능한 모든 언어의 자막을 다운로드하거나, 특정 언어의 자막만 다운로드 가능.
5. **프로그레스 훅** : 다운로드 진행 상황을 알 수 있는 프로그레스 훅을 제공. 이를 사용하여 다운로드 진행률을 출력하거나 다운로드 완료 시점을 알 수 있음.

1. yt-dlp

- 주요기능

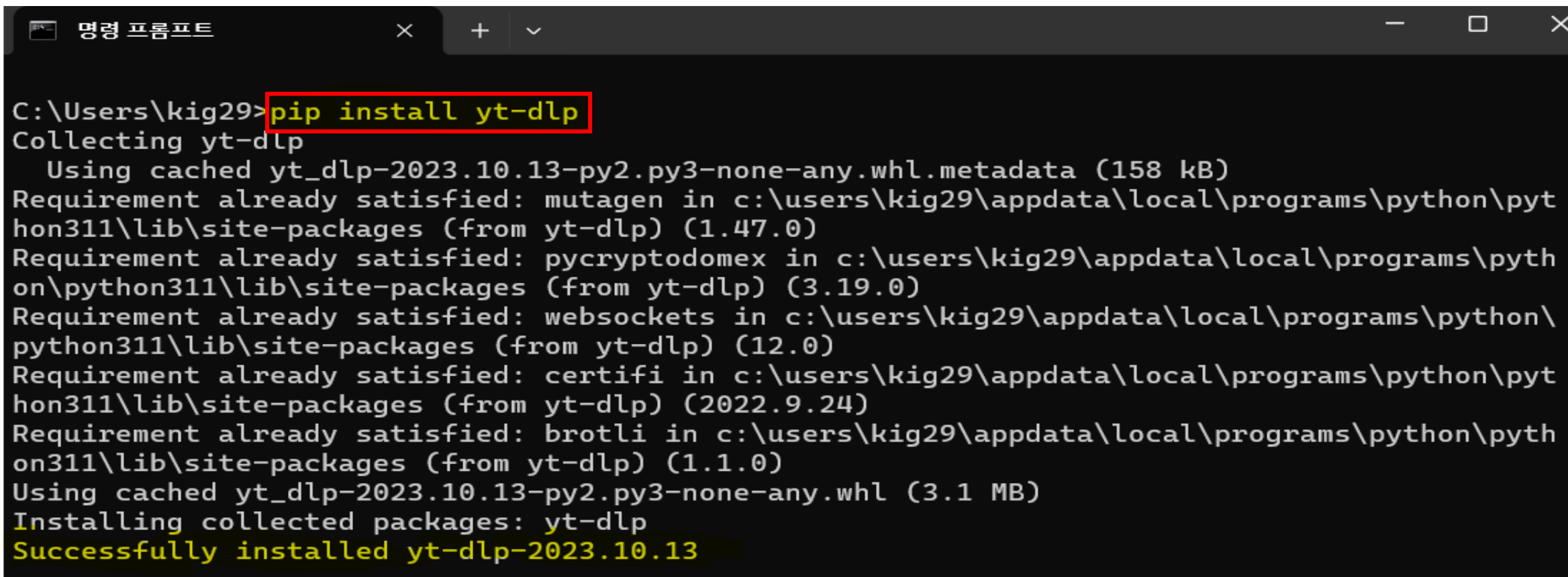
- 6. **플레이리스트 다운로드** : YouTube 플레이리스트의 모든 동영상을 한 번에 다운로드할 수 있음.

- yt-dlp는 명령줄 도구로서도 사용할 수 있지만, Python 코드 내에서 *YoutubeDL* 클래스를 사용하여 미디어를 다운로드하는 것이 더 유연하고 강력.

- *YoutubeDL* 클래스는 다양한 옵션을 설정할 수 있는 딕셔너리를 인자로 받아서, 원하는 방식으로 미디어를 다운로드할 수 있게 해줌.

1. yt-dlp

- yt-dlp 라이브러리 설치

A screenshot of a Windows Command Prompt window. The title bar at the top says '명령 프롬프트' (Command Prompt) and has standard window controls. The command prompt shows the execution of 'pip install yt-dlp' in a Windows directory 'C:\Users\kig29>'. The output shows that several dependencies (mutagen, pycryptodomex, websockets, certifi, brotli) are already satisfied. It then shows the installation of yt-dlp-2023.10.13, which is 3.1 MB. The final line indicates a successful installation.

```
C:\Users\kig29>pip install yt-dlp
Collecting yt-dlp
  Using cached yt_dlp-2023.10.13-py2.py3-none-any.whl.metadata (158 kB)
Requirement already satisfied: mutagen in c:\users\kig29\appdata\local\programs\python\python311\lib\site-packages (from yt-dlp) (1.47.0)
Requirement already satisfied: pycryptodomex in c:\users\kig29\appdata\local\programs\python\python311\lib\site-packages (from yt-dlp) (3.19.0)
Requirement already satisfied: websockets in c:\users\kig29\appdata\local\programs\python\python311\lib\site-packages (from yt-dlp) (12.0)
Requirement already satisfied: certifi in c:\users\kig29\appdata\local\programs\python\python311\lib\site-packages (from yt-dlp) (2022.9.24)
Requirement already satisfied: brotli in c:\users\kig29\appdata\local\programs\python\python311\lib\site-packages (from yt-dlp) (1.1.0)
Using cached yt_dlp-2023.10.13-py2.py3-none-any.whl (3.1 MB)
Installing collected packages: yt-dlp
Successfully installed yt-dlp-2023.10.13
```

2. 유튜브 동영상을 다운로드하는 기본 코드

```
1  import yt_dlp
2
3  ∨ def download_video(url):
4      |     ydl_opts = {}
5  ∨      |     with yt_dlp.YoutubeDL(ydl_opts) as ydl:
6      |         |     ydl.download([url])
7
8  ∨ def main():
9      |     url = input('Enter the website address to download : ')
10     |     download_video(url)
11
12  ∨ if __name__ == "__main__":
13     |     main()
14
```

2. 유튜브 동영상을 다운로드하는 기본 코드

행	설명
1	yt-dlp 모듈을 사용하려면 먼저 모듈을 설치해야 함. [pip install yt-dlp] GitHub : https://github.com/yt-dlp/yt-dlp
5	yt_dlp.YoutubeDL 객체를 생성하고 download 메소드를 호출하여 동영상을 다운로드함.
4	yd_opts는 다운로드 옵션을 설정하는 딕셔너리이며, YoutubeDL객체가 인자로 받음. 이 예제에서는 빈 딕셔너리를 사용하여 기본 옵션을 사용하여 다운로드 함.

- 위 코드는 동영상의 기본 형식을 다운로드 함. 다른 형식의 동영상을 다운로드하거나, 동영상에서 오디오만 추출하려면 ydl_opts를 적절히 설정. 자세한 내용은 깃허브의 문서를 참조.
- 또한, 이 코드는 유튜브의 서비스 이용 약관에 위배될 수 있습니다. 이 코드를 사용하여 동영상을 다운로드할 때는 **유튜브의 서비스 이용 약관을** 준수하십시오.

2. 유튜브 동영상을 다운로드하는 기본 코드

- 실행결과

```
Enter the website address to download : https://www.youtube.com/watch?v=[redacted]
[youtube] Extracting URL: https://www.youtube.com/watch?v=[redacted]
[youtube] HlN2BXNJzxA: Downloading webpage
[youtube] HlN2BXNJzxA: Downloading ios player API JSON
[youtube] HlN2BXNJzxA: Downloading android player API JSON
[youtube] HlN2BXNJzxA: Downloading m3u8 information
[info] HlN2BXNJzxA: Downloading 1 format(s): 22
[download] Destination: CHUNG HA [redacted].mp4
[download] 100% of 18.84MiB in 00:00:03 at 5.20MiB/s
```

3. ydl_opts 딕셔너리

```
ydl_opts = {  
    'format': 'bestaudio/best',  
    'outtmpl': '%(title)s.%(ext)s',  
    'restrictfilenames': True,  
    'nooverwrites': True,  
    'extractaudio': True,  
    'audioformat': 'mp3',  
    'audioquality': '192K',  
}
```

옵션	설명
format	다운로드할 동영상의 형식을 지정합니다. 예: 'bestaudio/best'
outtmpl	다운로드한 파일의 이름을 지정하는 템플릿입니다. 예: '%(title)s.%(ext)s'
restrictfilenames	True로 설정하면, 파일 이름에서 특수 문자를 제거하고 ASCII 문자만 유지합니다.
nooverwrites	True로 설정하면, 이미 존재하는 파일을 덮어쓰지 않습니다.
extractaudio	True로 설정하면, 동영상에서 오디오만 추출합니다.
audioformat	extractaudio가 True로 설정된 경우, 오디오 형식을 지정합니다. 예: 'mp3'
audioquality	오디오 품질을 설정합니다. 예: '192K'

- 이외에도 많은 옵션을 사용 가능.
- 모든 옵션에 대한 자세한 내용은 yt-dlp 문서를 참조

3. ydl_opts 딕셔너리

```
1 import yt_dlp
2
3 def download_video(url):
4     ydl_opts = {
5         'format': 'bestvideo[height<=1080]+bestaudio/best[height<=1080]'
6     }
7     with yt_dlp.YoutubeDL(ydl_opts) as ydl:
8         ydl.download([url])
9
10 def main():
11     url = input('Enter the website address to download : ')
12     download_video(url)
13
14 if __name__ == "__main__":
15     main()
```

- 1080p 이하의 최고 품질의 비디오 스트림과 최고품질의 오디오 스트림을 선택하는 설정.
- Yt-dlp는 비디오와 오디오 스트림을 별도로 다운로드한 후 합치는 방식을 사용하기 때문에, ffmpeg와 같은 외부 도구가 시스템에 설치되어 있어야 함.
- ffmpeg가 설치 되어 있지 않으면 아래와 같은 에러 메시지를 출력.

```
yt_dlp.utils.DownloadError: ERROR: You have requested merging of multiple formats but ffmpeg is not installed Aborting due to --abort-on-error
```

4. ffmpeg 설치 및 환경변수 등록

- ffmpeg : 오픈 소스의 멀티미디어 처리 라이브러리로, 비디오, 오디오, 자막 등 다양한 포맷을 처리할 수 있음.

1. Windows에서 ffmpeg 설치

- Ffmpeg 공식 홈페이지(<https://ffmpeg.org/download.html>)에서 Window용 바이너리를 다운로드
- 다운로드 받은 파일을 원하는 위치에 압축 해제 후, 환경 변수에 ffmpeg의 bin 디렉터리 경로를 추가

4. ffmpeg 설치 및 환경변수 등록

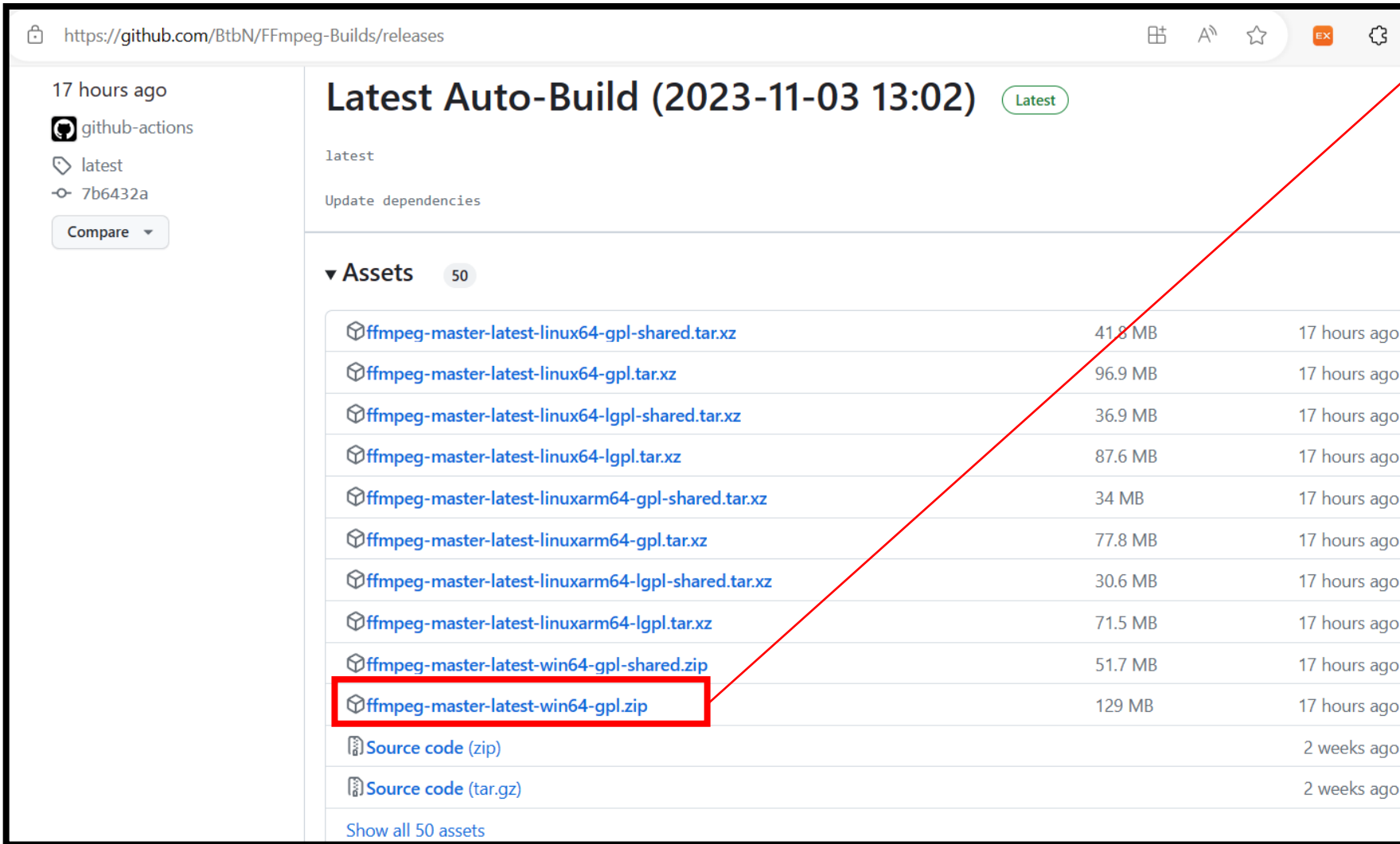
2. macOS에서 ffmpeg 설치

- Homebrew 패키지 관리자를 사용하여 쉽게 설치. 터미널에서 다음 명령을 실행.
- `Brew install ffmpeg`

3. Linux에서 ffmpeg 설치

- 우분투
- `sudo apt-get update`
- `sudo apt-get install ffmpeg`

4. ffmpeg 설치 및 환경변수 등록



https://github.com/BtbN/FFmpeg-Builds/releases

17 hours ago
github-actions
latest
7b6432a
Compare

Latest Auto-Build (2023-11-03 13:02) Latest

latest
Update dependencies

▼ Assets 50

ffmpeg-master-latest-linux64-gpl-shared.tar.xz	41.8 MB	17 hours ago
ffmpeg-master-latest-linux64-gpl.tar.xz	96.9 MB	17 hours ago
ffmpeg-master-latest-linux64-lgpl-shared.tar.xz	36.9 MB	17 hours ago
ffmpeg-master-latest-linux64-lgpl.tar.xz	87.6 MB	17 hours ago
ffmpeg-master-latest-linuxarm64-gpl-shared.tar.xz	34 MB	17 hours ago
ffmpeg-master-latest-linuxarm64-gpl.tar.xz	77.8 MB	17 hours ago
ffmpeg-master-latest-linuxarm64-lgpl-shared.tar.xz	30.6 MB	17 hours ago
ffmpeg-master-latest-linuxarm64-lgpl.tar.xz	71.5 MB	17 hours ago
ffmpeg-master-latest-win64-gpl-shared.zip	51.7 MB	17 hours ago
ffmpeg-master-latest-win64-gpl.zip	129 MB	17 hours ago
Source code (zip)		2 weeks ago
Source code (tar.gz)		2 weeks ago

Show all 50 assets

- 다운로드 받고 소스 코드가 있는 위치에 압축 해제
- 환경 변수에 ffmpeg의 bin 디렉토리 경로를 추가

4. ffmpeg 설치 및 환경변수 등록

- 파이썬을 이용한 환경변수 등록

set_env.py

```
1  import os
2
3  def path():
4
5      CURRENT_PATH = os.getcwd()
6      os.putenv('PATH', os.path.abspath(CURRENT_PATH) + '\\ffmpeg-master-latest-win64-gpl\\bin')
7      #print(os.environ.get('PATH'))
8
9  def main():
10     path()
11
12  if __name__ == "__main__":
13     main()
```

4. ffmpeg 설치 및 환경변수 등록

```
1  import yt_dlp, set_env
2
3  #ffmpeg 환경변수 등록 - 1080p 다운로드 시 필요
4  set_env.path()
5
6  def download_video(url):
7      ydl_opts = {
8          'format': 'bestvideo[height<=1080]+bestaudio/best[height<=1080]'
9      }
10     with yt_dlp.YoutubeDL(ydl_opts) as ydl:
11         ydl.download([url])
12
13     def main():
14         url = input('Enter the website address to download : ')
15         download_video(url)
16
17     if __name__ == "__main__":
18         main()
```

ytd.py

__pycache__

ffmpeg-master-latest-win64-gpl

set_env.py

ytd.py

4. ffmpeg 설치 및 환경변수 등록

- 실행결과

```
Enter the website address to download : https://www.youtube.com/watch?
[youtube] Extracting URL: https://www.youtube.com/watch?v=
[youtube] HlN2BXNJzxA: Downloading webpage
[youtube] HlN2BXNJzxA: Downloading ios player API JSON
[youtube] HlN2BXNJzxA: Downloading android player API JSON
[youtube] HlN2BXNJzxA: Downloading m3u8 information
[info] Testing format 616
[info] HlN2BXNJzxA: Downloading 1 format(s): 616+251
[hlsnative] Downloading m3u8 manifest
[hlsnative] Total fragments: 42
[download] Destination: CHUNG HA .mp4
[download] 100% of 66.58MiB in 00:00:17 at 3.84MiB/s
[download] Destination: CHUNG HA .webm
[download] 100% of 3.51MiB in 00:00:00 at 5.31MiB/s
[Merger] Merging formats into .webm"
Deleting original file .mp4 (pass -k to keep)
Deleting original file .webm (pass -k to keep)
```

궁금증??

MP4 vs WebM

- 위 소스 코드로 다운로드 받은 파일의 확장자가 WebM으로 MP4를 기대했던 내게는 WebM은 뭐지? 이런 궁금증에서 찾아봄.

항목	WebM	MP4
인코딩 방식	VP8 또는 VP9 비디오 코덱과 Vorbis 또는 Opus 오디오 코덱	H.264 (또는 AVC) 비디오 코덱과 AAC 오디오 코덱
호환성	HTML5 웹 브라우저에서 널리 지원되지만, 일부 장치나 미디어 플레이어에서는 호환되지 않을 수 있음	대부분의 장치와 미디어 플레이어에서 재생 가능
품질과 파일 크기	동일한 품질의 비디오를 더 작은 파일 크기로 압축 가능하지만 인코딩 시간이 더 길 수 있음	WebM보다 파일 크기가 크지만 인코딩 시간이 더 짧음
라이선스	완전히 오픈소스이고 무료로 사용 가능	H.264 코덱에 대한 특허료를 지불해야 하는 경우가 있음

5. 도전과제 - 1

- 1080p 이상의 동영상을 .MP4 형식으로 다운로드하는 소스코드를 작성.

```
1  import yt_dlp, set_env
2
3  #ffmpeg 환경변수 등록 - 1080p 다운로드 시 필요
4  set_env.path()
5
6  def download_video(url):
7      ydl_opts = {
8          'format': 'bestvideo[height>=1080][ext=mp4]+bestaudio[ext=m4a]/best[height>=1080][ext=mp4]',
9      }
10     with yt_dlp.YoutubeDL(ydl_opts) as ydl:
11         ydl.download([url])
12
13     def main():
14         url = input('Enter the website address to download : ')
15         download_video(url)
16
17     if __name__ == "__main__":
18         main()
```

5. 도전과제 - 2

- MP3 형식의 오디오를 다운로드하는 소스 코드 작성.

```
13 √ def download_audio(url):
14 √     ydl_opts = {
15         'format': 'bestaudio/best',
16 √         'postprocessors': [{
17             'key': 'FFmpegExtractAudio',
18             'preferredcodec': 'mp3',
19             'preferredquality': '192',
20         }],
21     }
22 √     with yt_dlp.YoutubeDL(ydl_opts) as ydl:
23         ydl.download([url])
```

- postprocessors 옵션은 다운로드한 파일에 대해 수행할 후처리 단계를 지정.
- FFmpegExtractAudio 후처리를 사용하여 오디오를 추출하고, 'mp3' 형식으로 변환
- Preferredquality 옵션을 사용하여 오디오 품질을 설정.

5. 도전과제 - 3

- 다운로드 파일의 위치와 이름을 지정할 수 있는 소스코드 작성.

```
'outtmpl': 'C:/downloads/%(title)s.%(ext)s',
```

- outtmpl 옵션을 사용하여 파일이 저장될 경로를 포함한 템플릿 문자열을 받음.

```
1  import yt_dlp, set_env
2
3  #ffmpeg 환경변수 등록 - 1080p 다운로드 시 필요
4  set_env.path()
5
6  def download_video(url):
7      ydl_opts = {
8          'format': 'bestvideo[height>=1080][ext=mp4]+bestaudio[ext=m4a]/best[height>=1080][ext=mp4]',
9          'outtmpl': './movie/%(title)s.%(ext)s',
10     }
11     with yt_dlp.YoutubeDL(ydl_opts) as ydl:
12         ydl.download([url])
13
14  def download_audio(url):
15      ydl_opts = {
16          'format': 'bestaudio/best',
17          'postprocessors': [{
18              'key': 'FFmpegExtractAudio',
19              'preferredcodec': 'mp3',
20              'preferredquality': '192',
21          }],
22          'outtmpl': './music/%(title)s.%(ext)s',
23     }
24     with yt_dlp.YoutubeDL(ydl_opts) as ydl:
25         ydl.download([url])
26
27  def main():
28      url = input('Enter the website address to download : ')
29      download_video(url)
30      #download_audio(url)
31
32  if __name__ == "__main__":
33      main()
```

5. 도전과제 - 4

- 다운로드 진행 상황을 확인하거나 다운로드 완료 시점을 알 수 있는 소스 코드 작성.
- YoutubeDL 클래스의 progress_hooks 옵션을 사용

```
6 def download_video(url):
7
8     def complete_status(down):
9         if down['status'] == 'finished':
10             print('\nDownload completed')
11
12     ydl_opts = {
13         'format': 'bestvideo[height>=1080][ext=mp4]+bestaudio[ext=m4a]/best[height>=1080][ext=mp4]',
14         'outtmpl': './movie/%(title)s.%(ext)s',
15         'progress_hooks': [complete_status], # 다운로드 진행 상황을 받아 처리하는 콜백함수
16     }
17     with yt_dlp.YoutubeDL(ydl_opts) as ydl:
18         ydl.download([url])
```

```

6  def download_video(url):
7
8      def complete_status(down):
9          if down['status'] == 'finished':
10             print('\nDownload completed')
11             if down['status'] == 'downloading':
12                 print(f"Downloading... {down['_percent_str']} complete")
13
14     ydl_opts = {
15         'format': 'bestvideo[height>=1080][ext=mp4]+bestaudio[ext=m4a]/best[height>=1080][ext=mp4]',
16         'outtmpl': './movie/%(title)s.%(ext)s',
17         'progress_hooks': [complete_status], # 다운로드 진행 상황을 받아 처리하는 콜백함수
18     }
19     with yt_dlp.YoutubeDL(ydl_opts) as ydl:
20         ydl.download([url])

```

- 다운로드 진행상황을 출력


```
1  import set_env
2  import yt_dlp
3
4
5  #ffmpeg 환경변수 등록 - 1080p 다운로드 시 필요
6  set_env.path()
7
8  def download_video(url):
9      def complete_status(down):
10         if down['status'] == 'downloading':
11             print(f"Downloading... {down['_percent_str']} complete")
12         elif down['status'] == 'finished':
13             print('\nDownload completed')
14
15         ydl_opts = {
16             'format': 'bestvideo[height>=1080][ext=mp4]+bestaudio[ext=m4a]/best[height>=1080][ext=mp4]',
17             'outtmpl': './movie/%(title)s.%(ext)s',
18             'progress_hooks': [complete_status], # 다운로드 진행 상황을 받아 처리하는 콜백함수
19         }
20         with yt_dlp.YoutubeDL(ydl_opts) as ydl:
21             ydl.download([url])
```

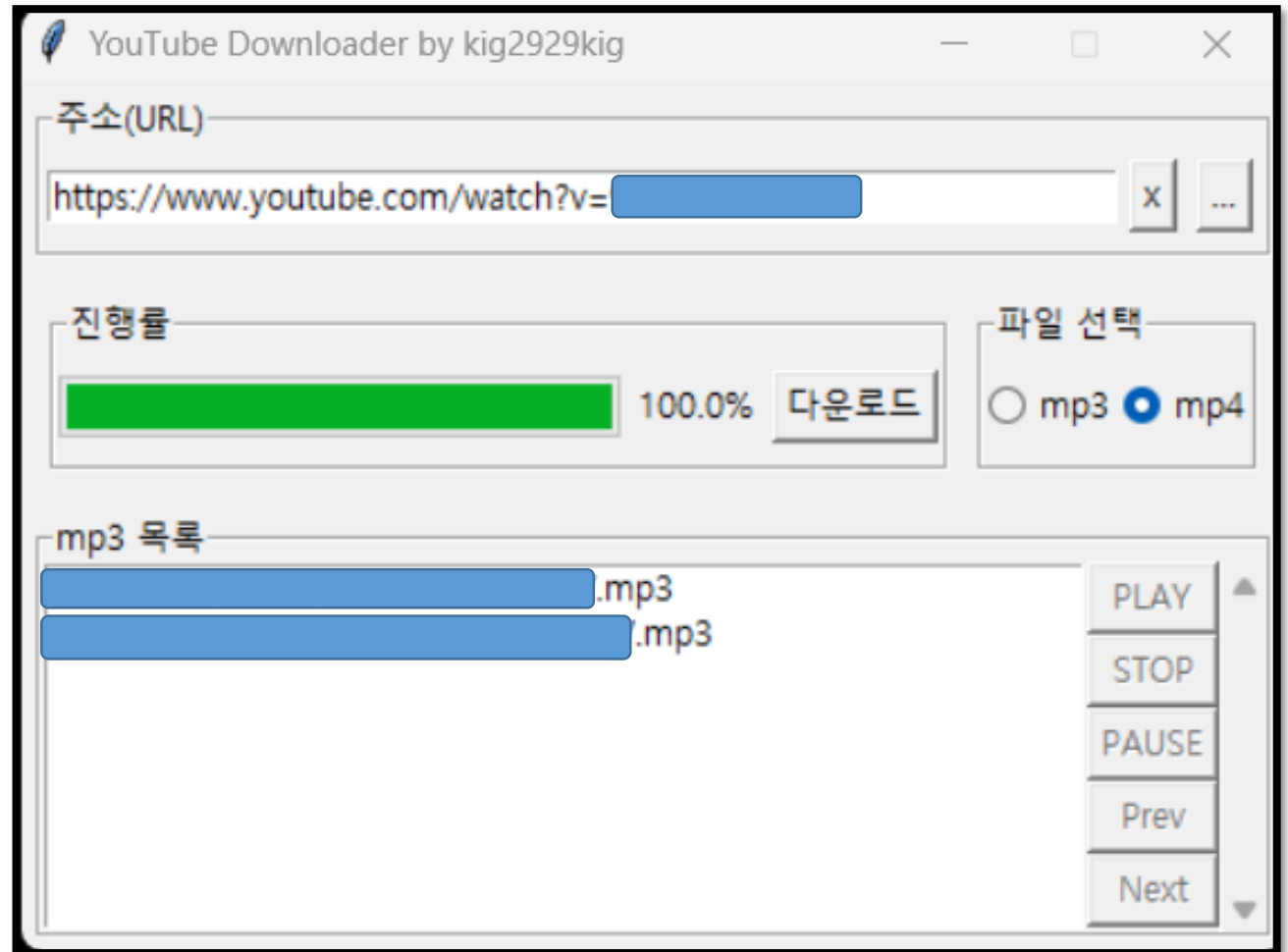
```

22
23 def download_audio(url):
24     def complete_status(down):
25         if down['status'] == 'downloading':
26             print(f"\nDownloading... {down['_percent_str']} complete")
27         elif down['status'] == 'finished':
28             print('\nDownload completed')
29
30     ydl_opts = {
31         'format': 'bestaudio/best',
32         'postprocessors': [{
33             'key': 'FFmpegExtractAudio',
34             'preferredcodec': 'mp3',
35             'preferredquality': '192',
36         }],
37         'outtmpl': './music/%(title)s.%(ext)s',
38         'progress_hooks': [complete_status], # 다운로드 진행 상황을 받아 처리하는 콜백함수
39     }
40     with yt_dlp.YoutubeDL(ydl_opts) as ydl:
41         ydl.download([url])
42
43 def main():
44     url = input('Enter the website address to download : ')
45     download_video(url)
46     #download_audio(url)
47
48 if __name__ == "__main__":
49     main()

```

5. 도전과제 - 5

- 유튜브 다운로더(GUI) 프로그램을 작성하시오.



참고

- <https://github.com/yt-dlp/yt-dlp>
- <https://wrtn.ai/>