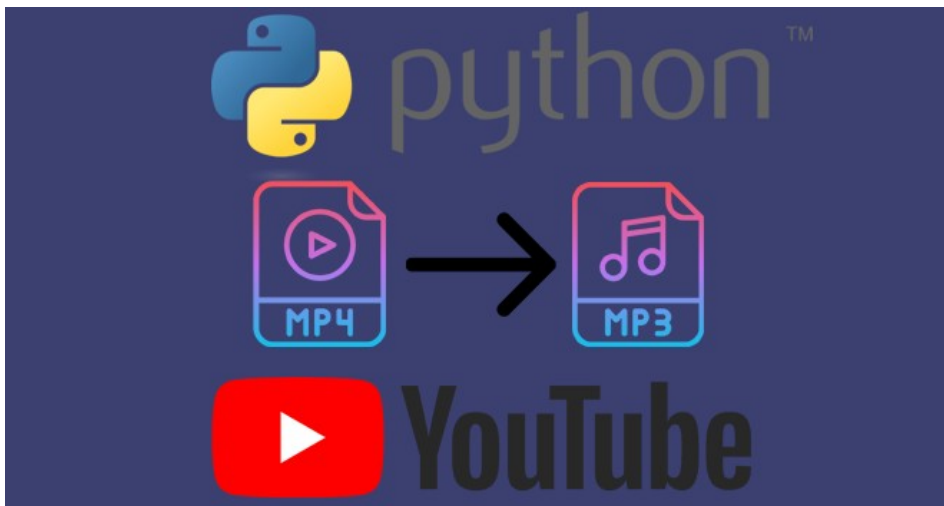


# Download Audio from YouTube: Convert MP4 to MP3 [Python + Pytube + FFmpeg]

<https://harshananayakkara.medium.com/download-audio-from-youtube-convert-mp4-to-mp3-python-pytube-ffmpeg-6163498c051f>

Download audio from any YouTube video of your choice and convert to MP3 format.

**UPDATE:** The Initial code has been updated to set a customized location — instead downloading the file to the same location where code resides — for the downloaded audio file. Therefore, the article also has been updated using the **UPDATE** keyword where necessary.



I usually like to download music or songs from YouTube, just the audio. However, I saw that many online downloaders download the audio in MP4 format. Conversely, I prefer all my audio files to be in MP3 format. Therefore, I thought to write a program that will help me to download and convert the audio files at once.

This code covers the below use-cases;

- This code is written on Windows machine so some steps may differ for other operating systems.
- Asks for user input to get the YouTube video URL.
- Initially, downloads the highest bit-rate audio stream in mp4.
- UPDATE:** Custom path can be set as the download location.
- Rename the downloaded file name.
- Convert the file to mp3.
- Delete the initially downloaded mp4 file after conversion.

## Prerequisites:

- Pytube: <https://pytube.io/en/latest/user/install.html>
- FFmpeg for Windows: <https://www.gyan.dev/ffmpeg/builds/ffmpeg-git-full.7z>

Now, let's dive into the code.

First of all we need to import the required libraries.

```
from pytube import YouTube
import os
```

**UPDATE:** Path variable added to customize the download location. For example, the following line will set the download path as `D:\Music\Mix\`.

```
path = 'D:\Music\Mix\\'
```

Then, we have to let the user to enter the YouTube URL and save it in `video_url` variable. After that, YouTube object is created named `yt`.

```
video_url = input("Please enter the video URL: ")
yt = YouTube(video_url)
```

Now, using the YouTube object we can get the highest bit-rate audio stream from the available codecs (defaults to mp4). Once it is selected, the file can be downloaded using the `download()` method.

**UPDATE:** Further, customized download path has been set using the `output_path` parameter.

```
audio = yt.streams.get_audio_only().audio.download(output_path=path)
```

Now the mp4 audio file should have been downloaded. Next, it is required to start the conversion. For that, we need the downloaded file name with extension.

```
file_name = audio.default_filename
```

**UPDATE:** Path and file name needs to be taken as an input for the next step, which is to rename the file to exclude any white spaces.

```
source = path + file_name
```

I have used FFmpeg to do the conversion. Therefore, for the program to work correctly no white spaces are allowed in the file name. So, I have replaced white spaces with underscore (`_`).

```
if ' ' in file_name:
    os.rename(source, source.replace(' ', '_'))
    file_name = source.replace(' ', '_')
```

`os.rename` replaces the white spaces with underscore in the downloaded mp4 file. `source.replace` will do the same thing but, will assign the new name to `file_name` variable for us to use later.

Up to now we have file name with mp4 extension. We also need to set the output file with mp3 format. For that, I have taken the same name without the extension. `os.path.splitext()` splits the extension and the rest of the file name. `os.path.splitext(file_name)[0]` will give us only the file name without the extension part.

```
file_without_ext = os.path.splitext(file_name)[0]
```

Now we have all parameters needed for the mp4 to mp3 conversion using FFmpeg.

*Note: `ffmpeg\bin` should be added to the Windows Environment Variable in order to the command to work.*

*Otherwise, it will give following error*

*'ffmpeg' is not recognized as an internal or external command, operable program or batch file.*

```
subprocess.Popen([ffmpeg, '-i', file_name, file_without_ext + '.mp3'])
```

Finally, we have to instruct the OS to run the above command and clean up the previously downloaded mp4 file.

```
os.system(command)os.remove(file_name)
```

Great! We have now completed the code. Let's see how it works.

```

PS D:\Project\youtube_mp3_downloader> python .\youtube_mp3.py
Please enter the video URL: https://www.youtube.com/watch?v=Cr2sGajaPx0
ffmpeg version 2022-10-02-git-5f02a261a2-full_build-www.gyan.dev Copyright (c) 2000-2022 the FFmpeg developers
built with gcc 12.1.0 (Rev2, Built by MSYS2 project)
configuration: --enable-gpl --enable-version3 --enable-static --disable-w32threads --disable-autodetect --enable-fontconfig --enable-icon
v --enable-gnutls --enable-libxml2 --enable-gmp --enable-bzlib --enable-lzma --enable-libsnap --enable-zlib --enable-librist --enable-lib
srt --enable-libssh --enable-libzmq --enable-avisynth --enable-libbluray --enable-libcaca --enable-sdl2 --enable-libaribb24 --enable-libdav
1d --enable-libdav1d --enable-libuavs3d --enable-libvbi --enable-libvrt --enable-libsvtav1 --enable-libwebp --enable-libx264 --enable-li
bx265 --enable-libxavs2 --enable-libxvid --enable-libaom --enable-libjxl --enable-libopenjpeg --enable-libvpx --enable-mediafoundation --en
able-libass --enable-frei0r --enable-libfreetype --enable-libfribidi --enable-liblensfun --enable-libvidstab --enable-libvmaf --enable-libz
img --enable-amf --enable-cuda-llvm --enable-cuvid --enable-ffnvcodec --enable-nvdec --enable-nvenc --enable-d3d11va --enable-dxva2 --enabl
e-libmfx --enable-libshaderc --enable-vulkan --enable-libplacebo --enable-opengl --enable-libcdio --enable-libgme --enable-libmodplug --ena
ble-libopenmpt --enable-libopencore-amrwb --enable-libmp3lame --enable-libshine --enable-libtheora --enable-libtwolame --enable-libvo-amrwb
enc --enable-libilbc --enable-libgsm --enable-libopencore-amrnb --enable-libopus --enable-libspeex --enable-libvorbis --enable-ladspa --ena
ble-libs2b --enable-libflite --enable-libmysofa --enable-librubberband --enable-libsoxr --enable-chromaprint
libavutil 57. 38.100 / 57. 38.100
libavcodec 59. 49.100 / 59. 49.100
libavformat 59. 33.100 / 59. 33.100
libavdevice 59.  8.101 / 59.  8.101
libavfilter  8. 49.100 /  8. 49.100
libswscale  6.  8.112 /  6.  8.112
libswresample 4.  9.100 /  4.  9.100
libpostproc 56.  7.100 / 56.  7.100
Input #0, mov,mp4,m4a,3gp,3g2,mj2, from 'D:\Music\Mix\Everybody Backstreets Back - Backstreet Boys (Lyrics) ..mp4':
Metadata:
  major_brand      : dash
  minor_version    : 0
  compatible_brands: iso6mp41
  creation_time    : 2022-07-08T19:15:13.000000Z
Duration: 00:04:07.29, start: 0.000000, bitrate: 129 kb/s
Stream #0:0[0x1](eng): Audio: aac (LC) (mp4a / 0x61347060), 44100 Hz, stereo, fltp, 5 kb/s (default)
Metadata:
  creation_time    : 2022-07-08T19:15:13.000000Z
  handler_name     : ISO Media file produced by Google Inc.
  vendor_id        : [0][0][0][0]
Stream mapping:
  Stream #0:0 -> #0:0 (aac (native) -> mp3 (libmp3lame))
Press [q] to stop, [?] for help
Output #0, mp3, to 'D:\Music\Mix\Everybody Backstreets Back - Backstreet Boys (Lyrics) ..mp3':

```

MP3 download output

```

Metadata:
  major_brand      : dash
  minor_version    : 0
  compatible_brands: iso6mp41
  TSSE             : Lavf59.33.100
Stream #0:0(eng): Audio: mp3, 44100 Hz, stereo, fltp (default)
Metadata:
  creation_time    : 2022-07-08T19:15:13.000000Z
  handler_name     : ISO Media file produced by Google Inc.
  vendor_id        : [0][0][0][0]
  encoder          : Lavc59.49.100 libmp3lame
size= 3865kB time=00:04:07.27 bitrate= 128.0kb/s speed=78.2x
video:0kB audio:3864kB subtitle:0kB other streams:0kB global headers:0kB muxing overhead: 0.008743%
PS D:\Project\youtube_mp3_downloader>

```

MP3 download output

Voila! Happy downloading 😊

Code in GitHub: [https://github.com/haarsh85/youtube\\_mp3\\_downloader](https://github.com/haarsh85/youtube_mp3_downloader)

I sincerely hope this article will be useful. I highly value your feedback and support!