protrolium / ffmpeg.md

Last active 2 days ago

using ffmpeg to extract audio from video files

ffmpeg.md

ffmpeg

Converting Audio into Different Formats / Sample Rates

Minimal example: transcode from MP3 to WMA:

```
ffmpeg -i input.mp3 output.wma
```

You can get the list of supported formats with:

```
ffmpeg -formats
```

Convert WAV to MP3, mix down to mono (use 1 audio channel), set bit rate to 64 kbps and sample rate to 22050 Hz:

```
ffmpeg -i input.wav -ac 1 -ab 64000 -ar 22050 output.mp3
```

Convert any MP3 file to WAV 16khz mono 16bit:

```
ffmpeg -i 111.mp3 -acodec pcm_s16le -ac 1 -ar 16000 out.wav
```

Convert any MP3 file to WAV 20khz mono 16bit for ADDAC WAV Player:

```
ffmpeg -i 111.mp3 -acodec pcm_s16le -ac 1 -ar 22050 out.wav
```

cd into dir for batch process:

```
for i in *.mp3; do ffmpeg -i "$i" -acodec pcm_s16le -ac 1 -ar 22050 "${i%.mp3}-encoded.wav"; done
```

Picking the 30 seconds fragment at an offset of 1 minute:
In seconds

```
ffmpeg -i input.mp3 -ss 60 -t 30 output.wav
```

In HH:MM:SS format

```
ffmpeg -i input.mp3 -ss 0:01:00 -t 0:00:30 output.wav
```

Extract Audio

```
ffmpeg -i video.mp4 -f mp3 -ab 192000 -vn music.mp3
```

The -i option in the above command is simple: it is the path to the input file. The second option -f mp3 tells ffmpeg that the ouput is in mp3 format. The third option i.e -ab 192000 tells ffmpeg that we want the output to be encoded at 192Kbps and -vn tells ffmpeg that we dont want video. The last param is the name of the output file.

You say you want to "extract audio from them (mp3 or ogg)". But what if the audio in the mp4 file is not one of those? you'd have to transcode anyway. So why not leave the audio format detection up to ffmpeg?

To convert one file:

```
ffmpeg -i videofile.mp4 -vn -acodec libvorbis audiofile.ogg
```

To convert many files:

```
for vid in *.mp4; do ffmpeg -i "$vid" -vn -acodec libvorbis "${vid%.mp4}.ogg"; done
```

You can of course select any ffmpeg parameters for audio encoding that you like, to set things like bitrate and so on.

Use `-acodec libmp3lame` and change the extension from `.ogg` to `.mp3` for mp3 encoding.

If what you want is to really extract the audio, you can simply "copy" the audio track to a file using -acodec copy. Of course, the main difference is that transcoding is slow and cpu-intensive, while copying is really quick as you're just moving bytes from one file to another. Here's how to copy just the audio track (assuming it's in mp3 format):

```
ffmpeg -i videofile.mp4 -vn -acodec copy audiofile.mp3
```

Note that in this case, the audiofile format has to be consistent with what the container has (i.e. if the audio is AAC format, you have to say audiofile.aac). You can use the ffprobe command to see which formats you have, this may provide some information:

```
for file in *; do ffprobe $file 2>&1 |grep Audio; done
```

A possible way to automatically parse the audio codec and name the audio file accordingly would be:

`for file in *mp4 *avi; do ffmpeg -i "$file" -vn -acodec copy "$file".` ffprobe "$file" 2>&1 |sed -rn 's/.*Audio: (...),* ./\1/p' `; done`

Note that this command uses sed to parse output from ffprobe for each file, it assumes a 3-letter audio codec name (e.g. mp3, ogg, aac) and will break with anything different.

Encoding multiple files

You can use a Bash "for loop" to encode all files in a directory:

```
$ mkdir newfiles
$ for f in *.m4a; do ffmpeg -i "$f" -codec:v copy -codec:a libmp3lame -q:a 2 newfiles/"${f%.m4a}.mp3"; done
```

```
ffmpeg -i input.m4a -acodec libmp3lame -ab 128k output.mp3
```
m4a to mp3 conversion with ffmpeg and lame

A batch file version of the same command would be:
```
for f in *.m4a; do ffmpeg -i "$f" -acodec libmp3lame -ab 256k "${f%.m4a}.mp3"; done
```

Merge Multiple Videos

file names in folder, if they contain spaces, must be properly escaped
```
ls * | perl -ne 'print "file $_"' | ffmpeg -f concat -i - -c copy merged.mp4
```

Split a Video into Images

```
$ ffmpeg -i video.flv image%d.jpg
```

Convert Images into a Video

```
$ ffmpeg -f image2 -i image%d.jpg imagestovideo.mpg
```

Convert mp4 to webm

```
$ ffmpeg -i example.mp4 -f webm -c:v libvpx -b:v 1M -acodec libvorbis example.webm -hide_banner
```
more info

Simple FLAC convert

```
ffmpeg -i audio.xxx -c:a flac audio.flac
```

Mix Stereo to Mono

You can modify a video file directly without having to re-encode the video stream. However the audio stream will have to be re-encoded.
Left channel to mono: `ffmpeg -i video.mp4 -map_channel 0.1.0 -c:v copy mono.mp4`

Left channel to stereo:
`ffmpeg -i video.mp4 -map_channel 0.1.0 -map_channel 0.1.0 -c:v copy stereo.mp4`

If you want to use the right channel, write `0.1.1` instead of `0.1.0`.

Trim End of file (mp3)

Here's a command line that will slice to 30 seconds without transcoding:
`ffmpeg -t 30 -i inputfile.mp3 -acodec copy outputfile.mp3`

To Encode or Re-encode ?

Do you need to cut video with re-encoding or without re-encoding mode? You can try to following below command.
Synopsis: ffmpeg -i [input_file] -ss [start_seconds] -t [duration_seconds] [output_file]

use FFmpeg cut mp4 video without re-encoding

Example:
`ffmpeg -i source.mp4 -ss 00:00:05 -t 00:00:10 -c copy cut_video.mp4`

use FFmpeg cut mp4 video with re-encoding

Example:
`ffmpeg -i source.mp4 -ss 00:00:05 -t 00:00:10 -async 1 -strict -2 cut_video.mp4`

If you want to cut off section from the beginning, simply drop -t 00:00:10 from the command

more commands
http://www.catswhocode.com/blog/19-ffmpeg-commands-for-all-needs

ishanjain28 commented on Sep 17, 2017

This is super useful. Thanks a lot. :)

jjgh commented on Sep 19, 2017

You're missing command lines from copying audio streams to other containers without re-encoding, which are among the most useful commands because you have no quality loss and can make a stream compatible with different devices! It's all in ffmpeg home page!

vahotm commented on Nov 11, 2017

ffmpeg -i input.mp4 a-vn -acodec copy output.aac
You're welcome.