Audio Processing - ffmpeg

Stefano Borzì

FF -> "fast forward"

MPEG -> Moving Picture Experts Group

Source: Wikipedia

FFmpeg – on GitHub

Product Solutions Open Source Pricing

Search or jump to...

FFmpeg

Overview Repositories 7 Projects Packages People 4

Popular repositories

FFmpeg Public
Mirror of https://git.ffmpeg.org/ffmpeg.git
C 39.8k 11.5k

web Public
Mirror of git://ffmpeg.org/ffmpeg-web
Less 242 106

nv-codec-headers Public
automatic mirror of https://git.videolan.org/?p=ffmpeg/nv-codec-headers.git
C 240 80

FFV1 Public
The FFV1 lossless video codec specification.
Shell 141 41

fateserver Public
Mirror of git://git.ffmpeg.org/fateserver
Perl 90 62

gas-preprocessor Public
Forked from yuvi/gas-preprocessor
Perl script that implements a subset of the GNU as preprocessor that Apple's as doesn't
Perl 67 35

People

Top languages
Perl C Shell Dockerfile Less

Install FFmpeg

$ sudo apt install ffmpeg

$ brew install ffmpeg

$ winget install --id=Gyan.FFmpeg -e

https://www.ffmpeg.org/download.html

$ ffmpeg -version

FFmpeg – codecs, formats

$ ffmpeg -formats

```
File formats:
 D. = Demuxing supported
 .E = Muxing supported
 --
 D  3dostr          3DO STR
  E 3g2             3GP2 (3GPP2 file format)
  E 3gp             3GP (3GPP file format)
 D  4xm             4X Technologies
  E a64             a64 - video for Commodore 64
 D  aa              Audible AA format files
 D  aac             raw ADTS AAC (Advanced Audio Coding)
 DE ac3             raw AC-3
 D  acm             Interplay ACM
```

[.....]

$ ffmpeg -codecs

```
Codecs:
 D..... = Decoding supported
 .E.... = Encoding supported
 ..V... = Video codec
 ..A... = Audio codec
 ..S... = Subtitle codec
 ...I.. = Intra frame-only codec
 ....L. = Lossy compression
 .....S = Lossless compression
 -------
 D.VI.S 012v                 Uncompressed 4:2:2 10-bit
 D.V.L. 4xm                  4X Movie
 D.VI.S 8bps                 QuickTime 8BPS video
 .EVIL. a64_multi            Multicolor charset for Commod
 .EVIL. a64_multi5           Multicolor charset for Commod
 D.V..S aasc                 Autodesk RLE
```

[.....]

Big Buck Bunny – free high quality video/audio

https://peach.blender.org/download/

https://download.blender.org/peach/bigbuckbunny_movies/

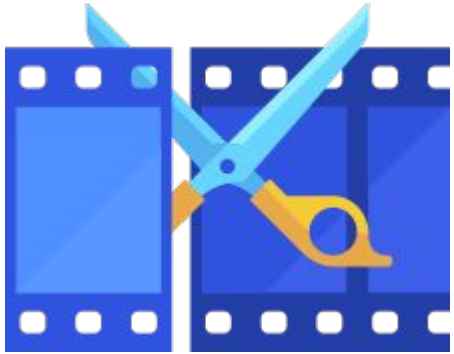
$ ffmpeg -i Big-Buck-Bunny.mp4

```
Input #0, mov,mp4,m4a,3gp,3g2,mj2, from 'Big-Buck-Bunny.mp4':
  Metadata:
    major_brand     : isom
    minor_version   : 512
    compatible_brands: isomiso2avc1mp41
    title           : Big Buck Bunny 60fps 4K - Official Blender Foundation Short Film
    artist          : Blender
    date            : 2014
    encoder         : Lavf60.3.100
    comment         : https://www.youtube.com/watch?v=aqz-KE-bpKQ
  Duration: 00:10:34.63, start: 0.000000, bitrate: 3382 kb/s
    Stream #0:0(und): Video: h264 (High) (avc1 / 0x31637661), yuv420p(tv, bt709),
1920x1080 [SAR 1:1 DAR 16:9], 3242 kb/s, 60 fps, 60 tbr, 15360 tbn, 120 tbc (default)
    Metadata:
      handler_name    : ISO Media file produced by Google Inc.
    Stream #0:1(und): Audio: aac (LC) (mp4a / 0x6134706D), 44100 Hz, stereo, fltp, 127
kb/s (default)
    Metadata:
      handler_name    : ISO Media file produced by Google Inc.
```

Cut video from time to time

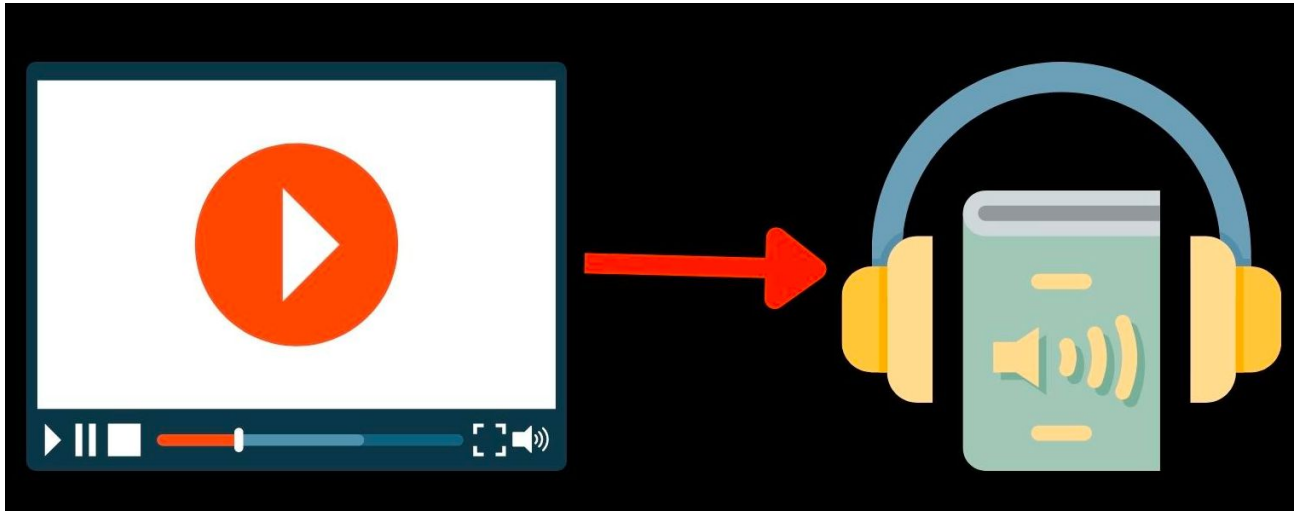
$ ffmpeg -ss **00:01:00** -to **00:02:00** -i **input.mp4** -c copy **output.mp4**

$ ffmpeg -ss **00:01:00** -t **60** -i **input.mp4 output.mp4**



Extract audio from video

```
$ ffmpeg -i output.mp4 -vn -acodec libmp3lame -ar 44100 -b:a 96k output.mp3
```



```
$ ffmpeg -i output.mp4 output.mp3
```

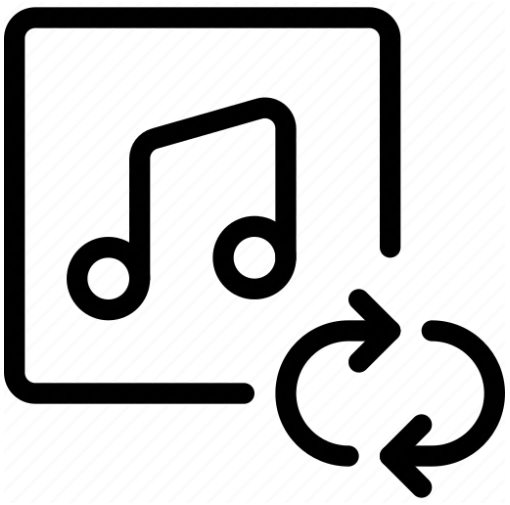
different quality and formats

```
$ ffmpeg -i output.mp4 -vn -acodec libmp3lame -ar 44100 -b:a 96k  output.mp3
$ ffmpeg -i output.mp4 -vn -acodec libmp3lame -ar 44100 -b:a 48k  output_2.mp3
$ ffmpeg -i output.mp4 -vn -acodec libmp3lame -ar 44100 -b:a 24k  output_3.mp3
```

```
$ ffmpeg -i output.mp3 output.wav
$ ffmpeg -i output.mp3 output.flac
```

Stream loop

```
$ ffmpeg -stream_loop 3 -i output.mp3 -c copy output-loop.mp3
```

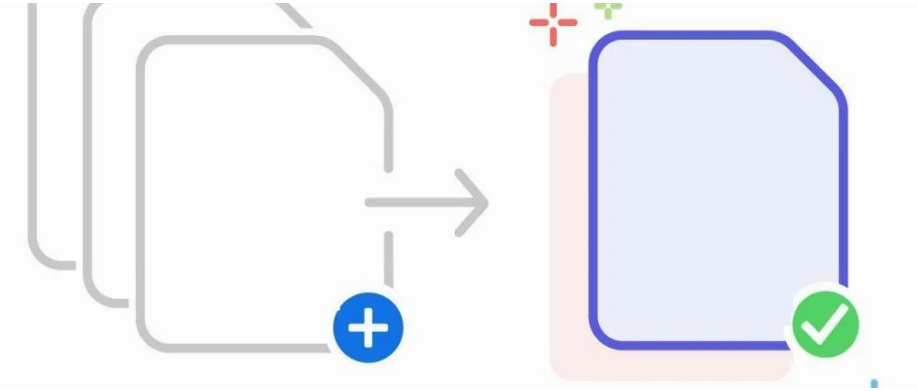


Merge more files

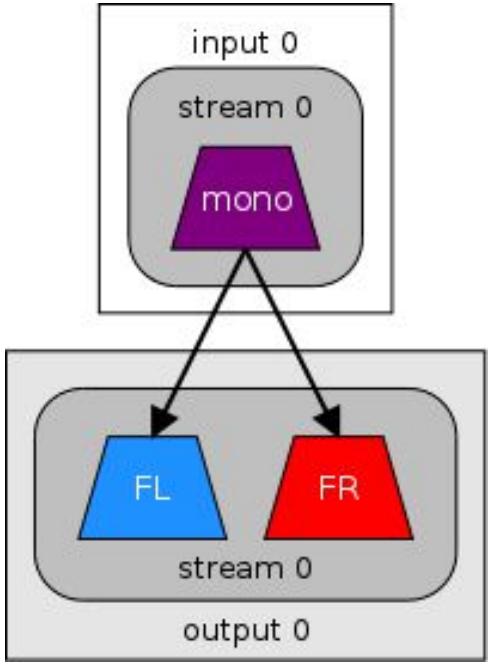
```
$ ffmpeg -f concat -i file.txt multiple-output.mp3
```

file.txt

file output.mp3
file output_2.mp3
file output_3.mp3

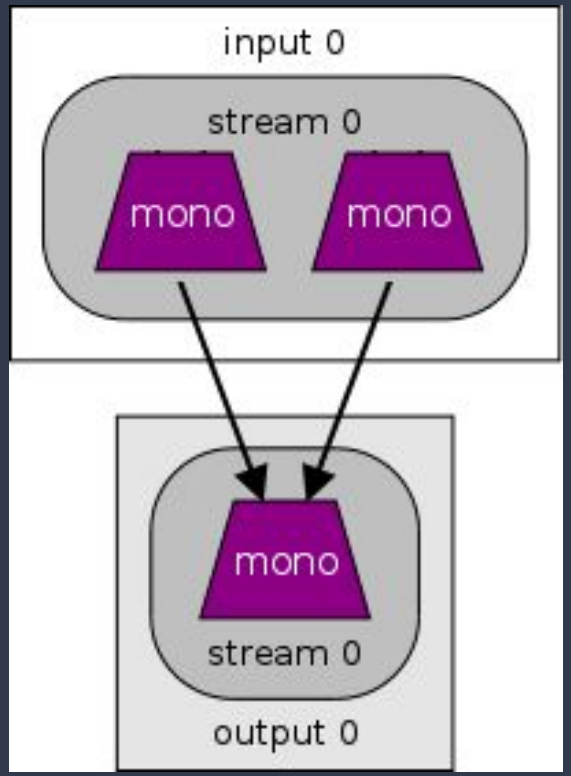


Make stereo from mono



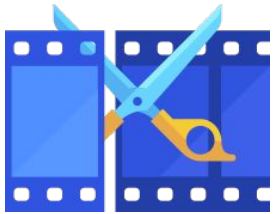
https://trac.ffmpeg.org/wiki/AudioChannelManipulation

Convert 2 channels into one




```
$ ffmpeg \
  -i output.mp3 \
  -ac 1 mono.mp3
```

Create FL and FR channels




```
$ ffmpeg \
  -ss 00:00:00 \
  -to 00:00:30 \
  -i mono.mp3 \
  -c copy output-30-1.mp3
```

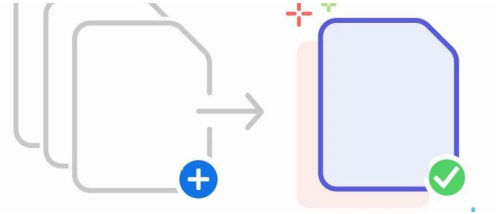
```
$ ffmpeg \
  -ss 00:00:30 \
  -to 00:01:00 \
  -i mono.mp3 \
  -c copy output-30-2.mp3
```

Create FL and FR channels


```
$ ffmpeg \
  -i output-30-1.mp3 \
  -af "pan=stereo|FL < c0" \
   LEFT-output.mp3
```


```
$ ffmpeg \
  -i output-30-2.mp3 \
  -af "pan=stereo|FR < c0" \
   RIGHT-output.mp3
```

Create FL and FR channels



```
$ ffmpeg -f concat -i file.txt stereo-LR.mp3
```

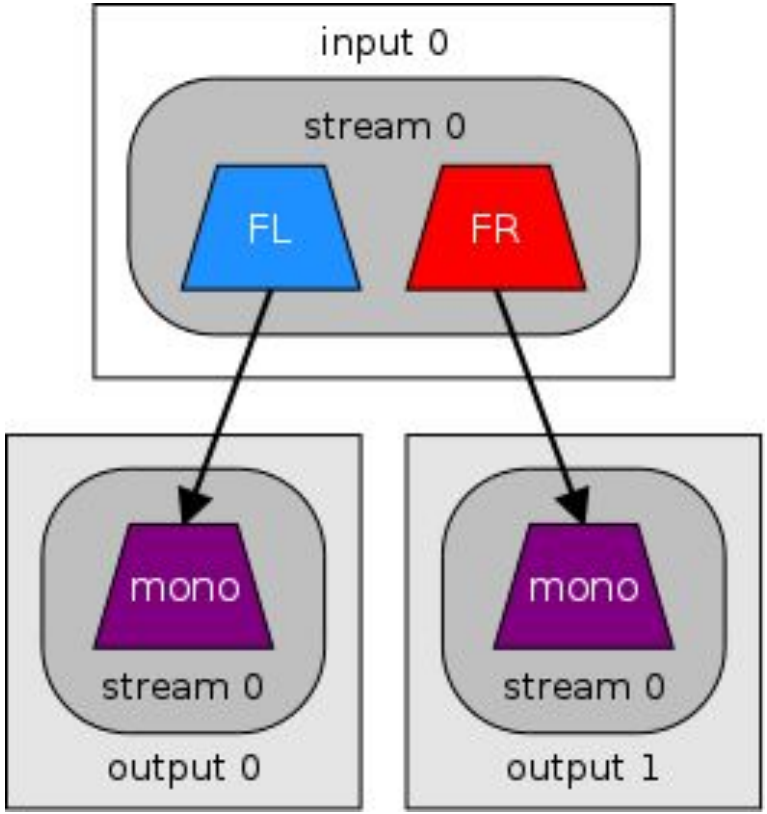
file.txt

file **LEFT-output**.mp3
file **RIGHT-output**.mp3

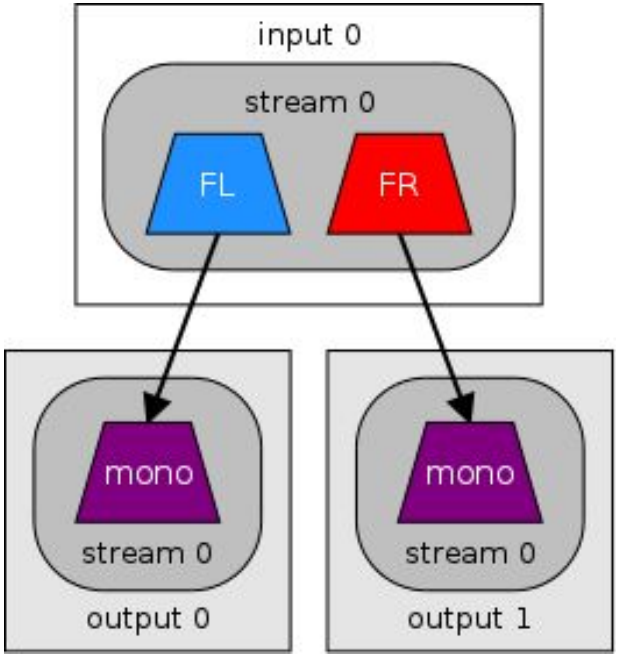
Stereo

Left - Right

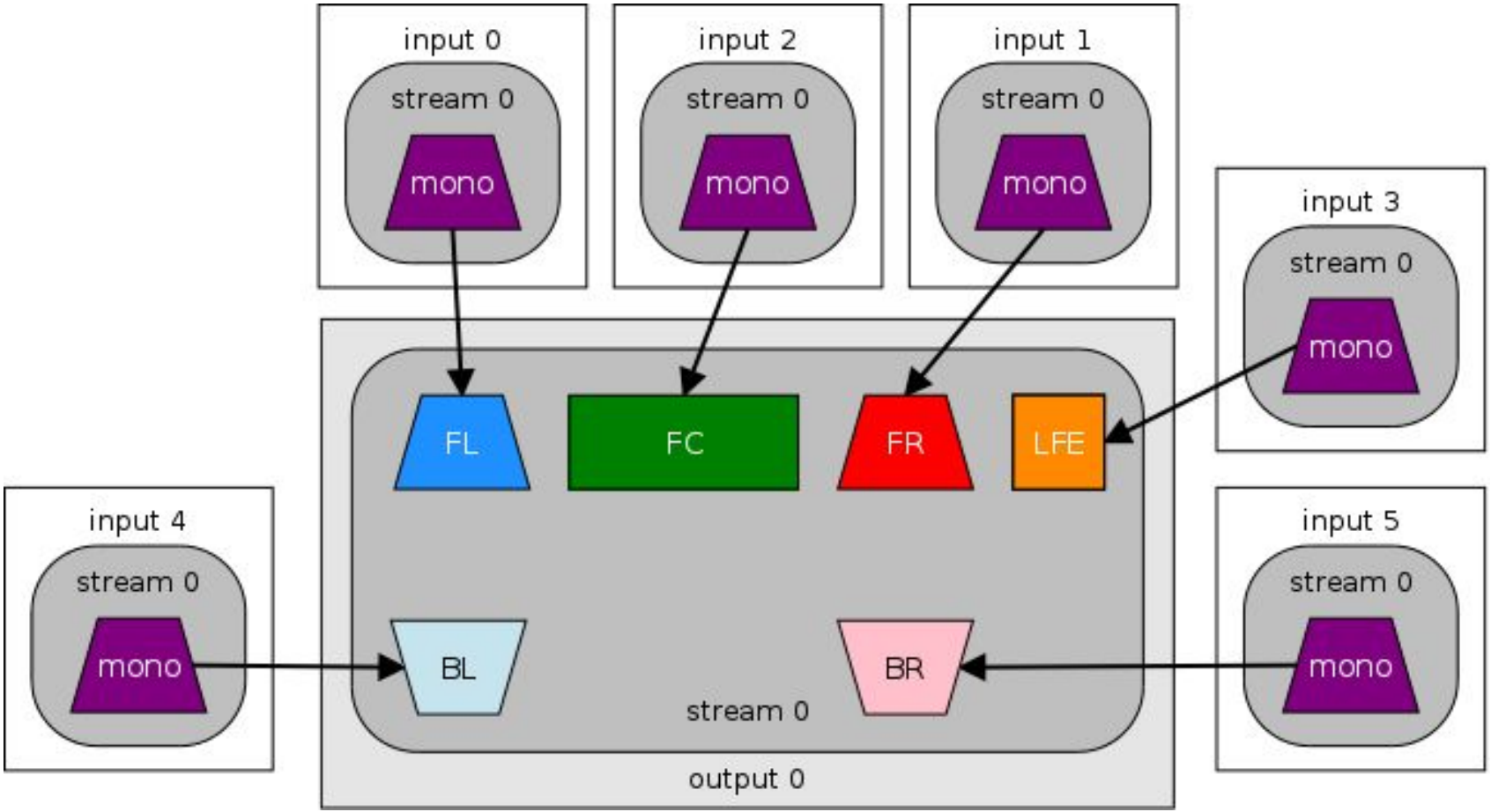
into 2 mono



Stereo -> Left - Right



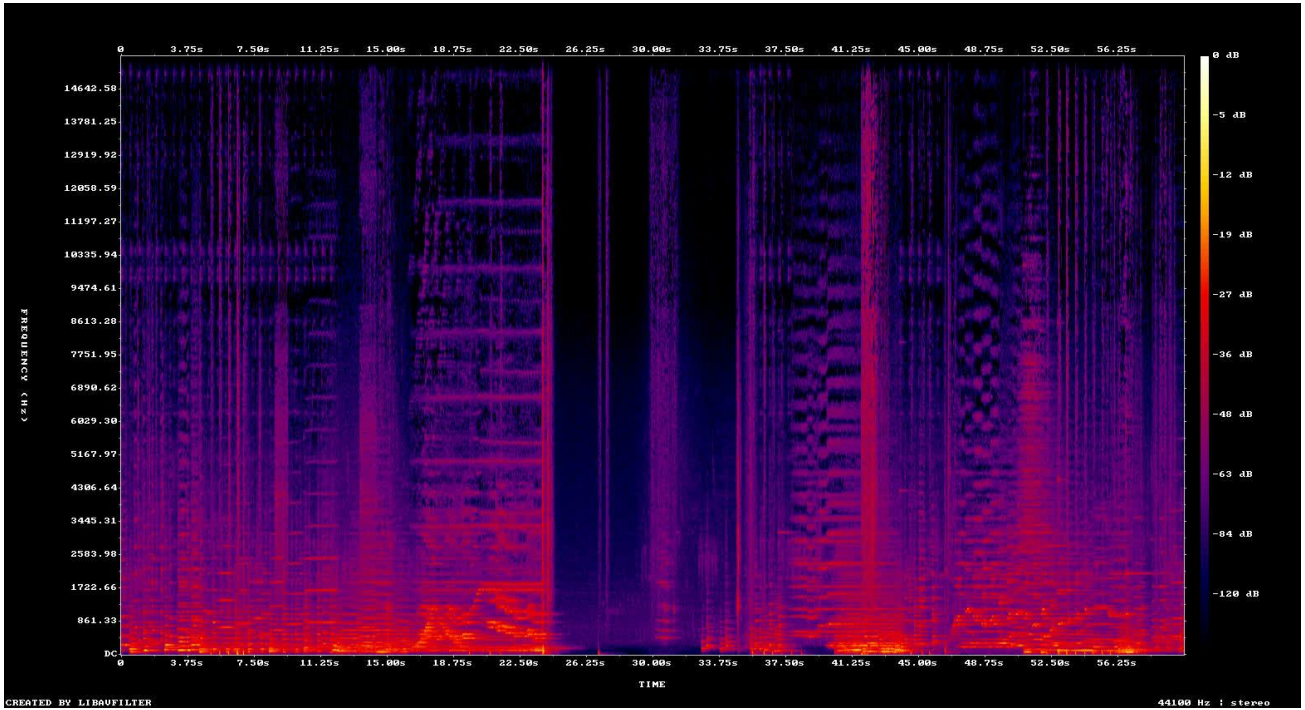

```
$ ffmpeg \
-i stereo-LR.mp3 \
-filter_complex
"[0:a]channelsplit=channel_layout=stereo[left][
right]" \
-map "[left]" left.mp3 \
-map "[right]" right.mp3
```


input 0
stream 0
mono
input 2
stream 0
mono
input 1
stream 0
mono
input 3
stream 0
mono
FL
FC
FR
LFE
input 4
stream 0
mono
BL
BR
stream 0
output 0
input 5
stream 0
mono

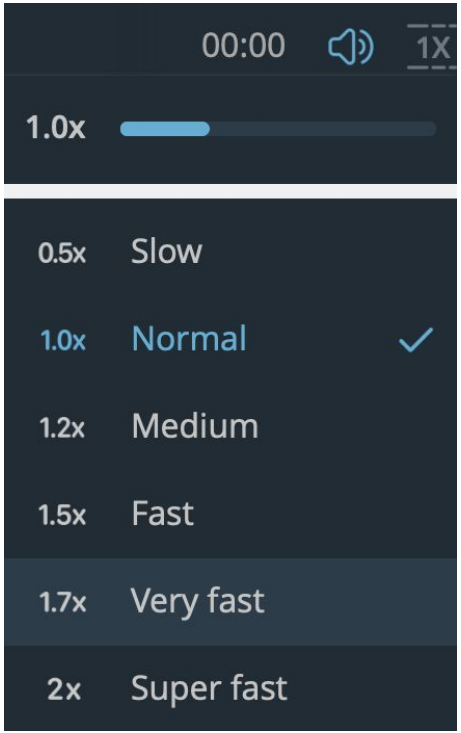

Speaker Testing for Apple USB audio device
Front Left
Test
Front Left-of-center
Test
Mono
Front Center
Test
Front Right-of-center
Test
Front Right
Test
Side Left
Test
Side Right
Test
Rear Left
Test
Rear Center
Test
Subwoofer
Test
Rear Right
Test
Close

FFmpeg – show spectrum

```
$ ffmpeg -i output.mp3 -lavfi showspectrumpic=s=hd720 out.jpg
```



FFmpeg – speeding up/slowing down audio



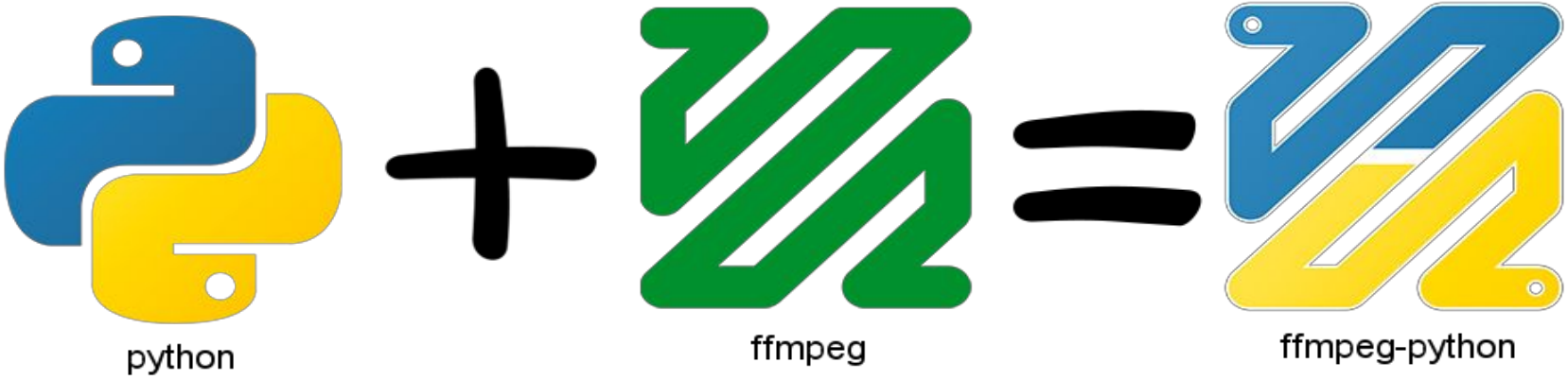

```
$ ffmpeg -i output.mp3 -filter:a "atempo=0.5" -vn output-0.5x.mp3
```

```
$ ffmpeg -i output.mp3 -filter:a "atempo=1.5" -vn output-1.5x.mp3
```

```
$ ffmpeg -i output.mp3 -filter:a "atempo=2.0" -vn output-2x.mp3
```

Python + FFmpeg

```
$ pip install ffmpeg-python
```



https://github.com/kkroening/ffmpeg-python

Python + FFmpeg

```
import subprocess

subprocess.run(['ffmpeg', '-i', ‘path/file.mp3’, '-q:a', 0, ‘path/output.mp3’], check=False)
```

Python + FFmpeg

Exercises:

- merge two audio files into one (overlay)
- concat multiple files
- Given two mono audio make one stereo file

https://github.com/kkroening/ffmpeg-python/blob/master/examples/README.md