

Preparing the weekly release
as one audio file



A complete, cross-platform solution to record,
convert and stream audio and video.

<https://ffmpeg.org/>

FFmpeg

FFmpeg is a command line tool for working with audio and video formats. You can use it to do a whole range of things, it comes with a lot of detailed settings and options.

This is an example of how you can use FFmpeg to convert a `mp4` video file into an `avi` file.

```
$ ffmpeg -i input.mp4 output.avi
```

You can use FFmpeg on the soupboat, where it is installed with `sudo apt install ffmpeg`; or you can install it on your own computer, in which case you can find installation instructions here:

- Mac: <https://formulae.brew.sh/formula/ffmpeg>
- Windows: <https://community.chocolatey.org/packages/ffmpeg>
- More options: <https://trac.ffmpeg.org/wiki/CompilationGuide>

When you are planning to FFmpeg often and specially to work with bigger files, it is recommended to install it on your own computer.

```
In [49]: ! ffprobe -hide_banner samples/owl.mp3
```

```
Input #0, mp3, from 'samples/owl.mp3':
  Metadata:
    comment           : dUBITS=$02021600
                     : dSCENE=1
                     : dTAKE=8
                     : dTAPE=20160224
                     : dFRAMERATE=24
                     : dTRK1=LEFT
                     :
    encoded_by        : iZotope RX 4
    originator_reference:
    USIZT0G88181ZVXYL092953326532331
    date              : 2016-03-05
    coding_history    :
    A=PCM,F=48000,W=24,M=1,T=ZaxConvert
                     :
    time_reference    : 3186656781
    encoder           : Lavf58.20.100
  Duration: 00:01:28.25, start: 0.023021, bitrate: 109
  kb/s
  Stream #0:0: Audio: mp3, 48000 Hz, stereo, fltp,
  109 kb/s
  Metadata:
    encoder           : Lavc58.35
```

```
In [50]: ! ffprobe -hide_banner samples/dialed.mp3
```

```
Input #0, mp3, from 'samples/dialed.mp3':
  Metadata:
    encoder           : Lavf58.20.100
  Duration: 00:00:12.02, start: 0.023021, bitrate: 128
  kb/s
  Stream #0:0: Audio: mp3, 48000 Hz, stereo, fltp,
  128 kb/s
  Metadata:
    encoder           : Lavc58.35
```

From wav to mp3

Many of the recording devices save audio as wav files. This is a high quality audio file format, which is too heavy to use on the web and most browsers will not play it in a webpage. \ So we will convert the audio files into a compressed audio format, that is supported by all browsers: mp3.

To convert a wav file into a mp3 file, you can use the following Ffmpeg command:

```
In [ ... ] ! ffmpeg -hide_banner -i samples/raphael.wav samples/raphael.mp3
```

```
Guessed Channel Layout for Input Stream #0.0 : stereo
Input #0, wav, from 'samples/raphael.wav':
  Duration: 00:00:16.39, bitrate: 1411 kb/s
  Stream #0:0 Audio: pcm_s16le ([1][0][0][0] / 0x0001), 44100 Hz, stereo, s16, 1411 kb/s
  Stream mapping:
    Stream #0:0 -> #0:0 (pcm_s16le (native) -> mp3 (libmp3lame))
  Press [q] to stop, [?] for help
  Output #0, mp3, to 'samples/raphael.mp3':
    Metadata:
      TSSE : Lavf58.20.100
      Stream #0:0 Audio: mp3 (libmp3lame), 44100 Hz, stereo, s16p
    Metadata:
      encoder : Lavc58.35.100 libmp3lame
      size= 257kB time=00:00:16.40 bitrate= 128.3kbits/s
      speed=18.2x
      video:0kB audio:257kB subtitle:0kB other streams:0kB
      global headers:0kB muxing overhead: 0.096236%
```

```
Input #0, concat, from 'playlist.txt':
  Duration: N/A, start: -0.023021, bitrate: 128 kb/s
  Stream #0:0 Audio: mp3, 48000 Hz, stereo, fltp, 128 kb/s
  Metadata:
    encoder : Lavc58.35
    Output #0, mp3, to 'release.mp3':
      Metadata:
        encoder : Lavc58.35
      Stream mapping:
        Stream #0:0 -> #0:0 (copy)
      Press [q] to stop, [?] for help
      size= 1630kB time=00:01:56.68 bitrate= 114.4kbits/s
      speed= 725x
      video:0kB audio:1629kB subtitle:0kB other streams:0kB
      global headers:0kB muxing overhead: 0.025712%
  Remove the file release.mp3 before you make a new one
  with:
```

```
In [52]: ! rm release.mp3
```

Check the format, encoder, bitrate, and channels of your audio file

```
In [54]: ! fftprobe -hide_banner samples/raphael.mp3
```

```
Input #0, mp3, from 'samples/raphael.mp3':
  Metadata:
    encoder : Lavf58.20.100
  Duration: 00:00:16.44, start: 0.023021, bitrate: 129 kb/s
  Stream #0:0 Audio: mp3, 48000 Hz, stereo, fltp, 129 kb/s
  Metadata:
    encoder : Lavc58.35
```

You can specify more options to convert your file with a specific codec or quality, set the number of channels and select a bit rate. \ See the command below, and this page for more information: <https://trac.ffmpeg.org/wiki/Encode/MP3>

```
In... ! ffmpeg -i samples/raphael.wav -acodec libmp3lame -aq 2 -ac 2 -ar 48000 -hide_banner samples/raphael.mp3
```

- `-acodec` : name of the audio codec
- `-aq` : quality scale (1-10)
- `-ac` : number of audio channels
- `-ar` : frequency in hz
- `-ab` : audio bitrate

See: <https://ffmpeg.org/ffmpeg.html#Audio-Options>

From aiff to mp3

```
In [ ... ! ffmpeg -hide_banner -i samples/dialed.aiff samples/dialed.mp3
```

Hide "WARNING: library configuration mismatch"

Add `-hide_banner` to your `ffmpeg` commands, to avoid getting the `[warning]` messages in your output.

```
In [ ]: ! ffmpeg -hide_banner
```

Generating playlist.txt

To start making a single audio file of all the recordings, we will first make a `playlist.txt` file.

To do this, we will use the concatenate function of FFmpeg. The following examples come from this page: <https://trac.ffmpeg.org/wiki/Concatenate>

We will only search for all the `mp3` files in the `./samples/` folder:

```
In [2... ! for f in ./samples/*.mp3; do echo "file '$f'" >> playlist.txt; done
```

See if it worked:

```
In [29]: ! cat playlist.txt  
file './samples/dialed.mp3'  
file './samples/owl.mp3'  
file './samples/raphael.mp3'
```

Remove the file `playlist.txt` if you want to overwrite it and make a whole new one with:

```
In [26]: ! rm playlist.txt
```

Concatenating release.mp3

Now we will use the `playlist.txt` file to generate a single audio file, using FFmpeg's `concat`, which is the command for *concatenating*.

```
In [... ! ffmpeg -hide_banner -f concat -safe 0 -i playlist.txt -c copy release.mp3
```