

Videokompression mit FFmpeg / ffmpeg4



Diese Anleitung richtet sich an *technisch Interessierte* mit dem Problem, dass Videos direkt ab Mobiltelefon und Digitalkamera oft *riesengroß* (Dateigröße) sind. Mit den entsprechenden Werkzeugen ist es möglich, die Dateigröße der Videos zu verringern.

Um den Praxiswert zu erhöhen, verwende ich *Verein-fachungen*, also: wer mehr weiß, Geduld...

Die Videodatei

Technisch ist eine Videodatei ein *Container* (Behälter), der eine Videospur und eine Audiospur enthält. Man erkennt das Format dieses Containers an der Dateiendung, z.B. .AVI; .MOV, MP4. Darin werden sowohl Video als auch Audio mit bestimmten Verfahren (*Codec*) komprimiert gespeichert.

Je nach Containerformat können darin zusätzliche Informationen enthalten sein, wie Untertitel, Kapitel, weitere Audiospuren, usw. In der letzten Zeit haben sich die Container MP4 und MKV etabliert.

Kompression

Moderne Codecs (Smartphone, TV, Camcorder) arbeiten meist verlustbehaftet. Man sucht deshalb bei der Aufzeichnung einen gangbaren Kompromiss: *hohe Bildqualität* ↔ *geringe Dateigröße* (bedingt hohe Prozessorleistung).

Da der Prozessorleistung bei tragbaren Geräten Grenzen gesetzt sind, sind deren Videodateien oft riesengroß. Die Lösung aus diesem Dilemma ist die Transkodierung, d.h. das erneute Komprimieren mit ausgewählten Einstellungen (Container, Codecs, Bildformat/größe, Kompressionsrate) auf einem leistungsfähigen Computer.

Die meisten kostenlosen Video-Kodierer basieren auf FFmpeg, einer Open Source Lösung, welche eine Unzahl von Codecs beherrscht. FFmpeg selbst ist ein recht anspruchsvolles Kommandozeilen-Programm. Deshalb habe ich ffmpeg4 geschrieben. Es verzichtet auf allen Schnickschnack, sodass auch Ungeübte damit klarkommen. Für höhere Ansprüche empfehle ich HandBrake.

Transkodierung mit ffmpeg4

Start | ffmpeg4 | ffmpeg4 - FFmpeg-GUI Enter

In drei Schritten

1. **Quellordner** : zu komprimierenden Dateien.
2. **Zielordner** : resultierende Dateien.
3. **FFmpeg-Aufgabe** : Einstellungen für den Kompressionsvorgang

FFmpeg-Aufgabe

- MP4**: ideal für die meisten Anwendungen, hohe Kompatibilität
- MKV**: freies Format (ermöglicht Untertitel, zweite Audiospur usw.)
- Auflösung**: HD720 entspricht dem HD-TV-Standard (2024)
- Voreinstellung**: Effizienz des Encoders (ultrafast → größere Dateien)
- 16:9** ergänzt das Video ggf. mit schwarzen Balken (TV-Aspekt)
- Schnitt/Extras**: Startzeit und/oder Dauer im Format **hh:mm:ss**.^{nm}/₁₀₀
Alternativ: FFmpeg-Argumente, z.B. `-r 5` oder `-b:a 48k`
- Test: 15s** Komprimiert 15s zu Testzwecken und spielt sie mit VLC ab.
- Nur Audio**: M4A oder MP3 für die Audio-Extraktion.

Tipp:

Probieren geht über Studieren...

Problemen, Fragen?

bigler.thomas@gmail.com


FFmpeg arbeitet – Geduld...

Die verlustbehaftete Datenkompression ist rechenintensiver Vorgang und dauert entsprechend lange:

```
FFmpeg: Konsole arbeitet ([Q] für Abbruch)
Program 9212
Metadata:
  service_name      : ?ITV HD
  service_provider: upc
Stream mapping:
  Stream #0:0 -> #0:0 (h264 (native) -> h264 (libx264))
  Stream #0:1 -> #0:1 (mp2 (native) -> aac (native))
Press [q] to stop, [?] for help
[libx264 @ 000001804121f980] using SAR=1/1
[libx264 @ 000001804121f980] using cpu capabilities: MMX2 SSE2Fast SSSE3 SSE4.2 AVX XOP FMA3 BMI1
[libx264 @ 000001804121f980] profile High, level 3.2
[libx264 @ 000001804121f980] 264 - core 155 r2893 b00bcaf - H.264/MPEG-4 AVC codec - Copyleft 2003-2017 - http://www.vid
eolan.org/x264.html - options: cabac=1 ref=3 deblock=1:0:0 analyse=0x3:0x113 me=hex subme=7 psy=1 psy_rd=1.00:0.00 mixed
_ref=1 me_range=16 chroma_me=1 trellis=1 8x8dct=1 cqm=0 deadzone=21,11 fast_pskip=1 chroma_qp_offset=-2 threads=9 lookah
ead_threads=1 sliced_threads=0 nr=0 decimate=1 interlaced=0 bluray_compat=0 constrained_intra=0 bframes=3 b_pyramid=2 b_
adapt=1 b_bias=0 direct=1 weightb=1 open_gop=0 weightp=2 keyint=250 keyint_min=25 scenecut=40 intra_refresh=0 rc_lookahe
ad=40 rc=crf mbtree=1 crf=23.0 qcomp=0.60 qpmin=0 qpmax=69 qpstep=4 ip_ratio=1.40 aq=1:1.00
Output #0, mp4, to 'C:\Users\Thomas\Desktop\NEU-Großes Video 1.mp4':
Metadata:
  encoder           : Lavf58.6.100
  Stream #0:0: Video: h264 (libx264) (avc1 / 0x31637661), yuv420p, 1280x720 [SAR 1:1 DAR 16:9], q=-1--1, 50 fps, 12800
  tbn, 50 tbc
Metadata:
  encoder           : Lavc58.9.100 libx264
Side data:
  cpb: bitrate max/min/avg: 0/0/0 buffer size: 0 vbv_delay: -1
  Stream #0:1(deu): Audio: aac (LC) (mp4a / 0x6134706D), 48000 Hz, stereo, fltp, 64 kb/s
Metadata:
  encoder           : Lavc58.9.100 aac
frame= 379 fps= 67 q=31.0 size= 768kB time=00:00:06.91 bitrate= 910.3kbits/s dup=65 drop=0 speed=1.22x
```

Farblich hervorgehoben sind die für uns nützlichen Informationen:

- 1 Videocodec hier **libx264**, der TV-Standard, wird von den meisten aktuellen Geräten dekodiert.
- 2 Audiocodec hier **aac**, ([Advanced Audio Coding](#)), wird von den meisten aktuellen Geräten dekodiert.
- 3 Speed Kompressionsgeschwindigkeit im Verhältnis zur normalen Abspieldauer.
Hier: Die Kompression erfolgt 1.22 × schneller als das Abspielen.

 **ffmpeg** liest alle gängigen Video- und Audioformate. Es erlaubt die Stapelverarbeitung der ausgewählten Dateien in der Liste (links). Dabei gelten die Einstellungen für sämtliche zu bearbeitenden Dateien.

Kompressionsrate

Im vorliegenden Beispiel (mp4, H.264/aac) ist die transkodierte Datei **5.2 × kleiner** als das Original. Dabei sind kaum [Kompressions-Artefakte](#) feststellbar. Es kann sich lohnen – abhängig von Quellmaterial und Einsatzzweck – eine kleinere Bildformat (**HD576** oder **HD480**) zu wählen oder auch die **HQ**-Variante auszuprobieren.

Hinweis: Während der Kodierung gibt FFmpeg viele Details zur Kompression preis. Was für den Laien wie schreckliche Fehler aussehen mag, sind in Wirklichkeit *diagnostische Hinweise*, welche man bei zufriedenstellendem Resultat (Zieldatei unbedingt *kontrollieren!*) getrost ignorieren kann.

Ziel erreicht

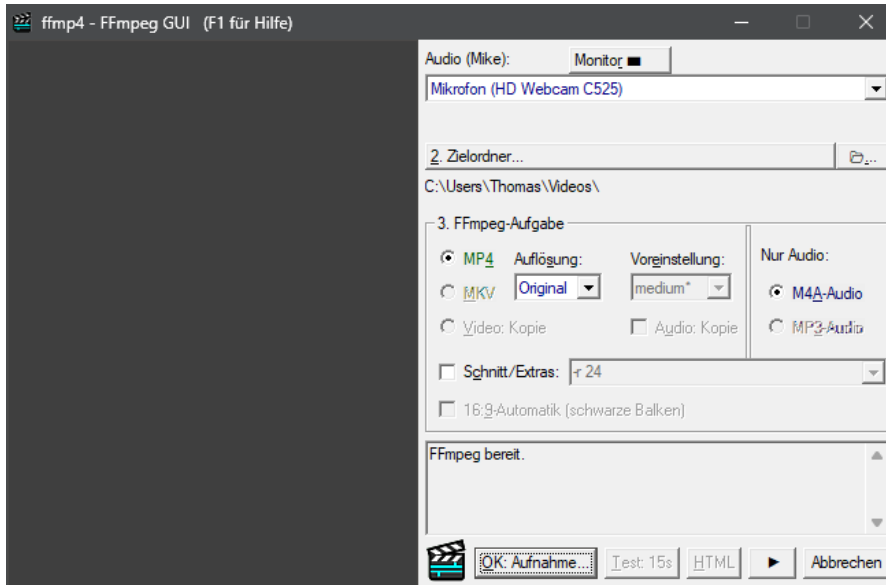
Die *transkodierten* Dateien erhalten im Dateinamen das Präfix **_NEU-**Alter Dateiname.mp4, so kann man sie leicht von den Originalen unterscheiden. Unbedingt die Originale aufbewahren (z.B. auf einem externen Datenträger). So ist sichergestellt, dass man ggf. nachträglich verlustfrei Änderungen (Schnitt, Effekte o.ä.) vornehmen kann.

Schnitt/Extras

Unter **Schnitt/Extras** bietet  **ffmpeg** zusätzliche Kodier-Optionen (alle Kommentare nach **¶** werden ignoriert):

- | | |
|--|---|
| -crf 30 -r 16 ¶ Video LQ, Bildrate 16 | -crf 30 Standard = 23, größerer Wert → kleinere Dateien, aber u. U. mit Artefakten (ausprobieren...) |
| | -r 16 Standard = 25, nach Bedarf verkleinern, erlaubt sehr kleine Dateien |
| -b:a 56k -af "dynaudnorm=f=150" ¶ Telefon | Sprache (kleine Bitrate, dynamische Kompression) |
| MKV→MP4 ¶ Format-Umwandlung | Wandelt in das jeweils andere Containerformat um. |

Bildschirmaufnahme (Screen Capture)



- 1. Monitor :**
Hier die Audio Quelle wählen (Mikrofon)
- 2. Zielordner:**
resultierende Dateien.
- 3. FFmpeg-Aufgabe:**
ggf. die Auflösung anpassen
- 4. Schnitt/Extra:**
Für Präsentationen die Bildfrequenz heruntersetzen, z.B. **-r 8**
→ kleinere Videodateien

Bildschirmaufnahme

Manchmal möchte man ein Verfahren oder einen Vorgang am PC dokumentieren und mit Audio-Kommentar versehen – auch dafür eignet sich **ffmp4**. Durch einen Klick auf **Monitor** schaltet das Programm in den Bildschirmaufnahme-Modus.

Tonmitschnitt

ffmp4 versucht, einen gültigen Mikrofoneingang zu erkennen. Je nach PC muss man hier etwas nachhelfen...

Falls kein gültiger *Mikrofoneingang* verfügbar ist, kann man auch **<kein Audio-Mitschnitt>** anwählen. Bis auf die Auflösung werden alle Parameter automatisch auf «vernünftige» Werte gesetzt. Für Videos auf Videoplattformen lohnt es sich u.U., die Bildschirmauflösung des PCs vorübergehend herunterzusetzen. So fällt es dem Publikum leichter, sich auf komplexen Bildern zu orientieren.

Den Aufnahmepegel des Mikrofons *vor dem Aufnehmen* unbedingt auf einen brauchbaren Wert zu setzen (rechte Maustaste auf das Lautsprecher-Symbol, **Soundeinstellungen öffnen**): genügend hohe Lautstärke, keine Verzerrungen. Die eigentliche Aufnahme wird mit **OK: Aufnahme** gestartet. *Wichtig:* Zum *Beenden* im blauen FFmpeg-Konsolenfenster die Taste **Q** drücken.

Die resultierende Datei (**2021-02-23@10-18-36-ScreenCapture.mp4**) kann selbstverständlich nachbearbeitet oder geschnitten werden. Durch einen erneuten Klick auf **Monitor** schaltet **ffmp4** wieder in den normalen Modus zurück.

Videos & Audios präsentieren

Damit deine Medien zuverlässig auf Fremd-Computern laufen, kann **ffmp4** HTML-basierte Media-Player erstellen. Klicke dazu auf den **HTML** Knopf. Im Quellverzeichnis werden nun die beiden Dateien **_START.cmd** und **_video.html** resp. **_audio.html** erstellt. Sie erlauben eine komfortable Wiedergabe direkt im Webbrowser.

Achtung: Browser können nur Videos im mp4-Format (Video: mp4/H.264 / Audio: m4a) abspielen. Vorhandene Untertitel im Format **WebVTT** werden mit in **_video.html** eingebunden (ideal für den Sprachunterricht). Der so erstellte Ordner kann auch auf einem Webserver verwendet werden. Die Datei **_START.cmd** erlaubt die *lokale Medienwiedergabe* in modernen Browsern. Für Audiodateien wird die Webseite **_audio.html** erstellt. Browser können Audios in den Formaten mp3 / m4a und ogg wiedergeben.

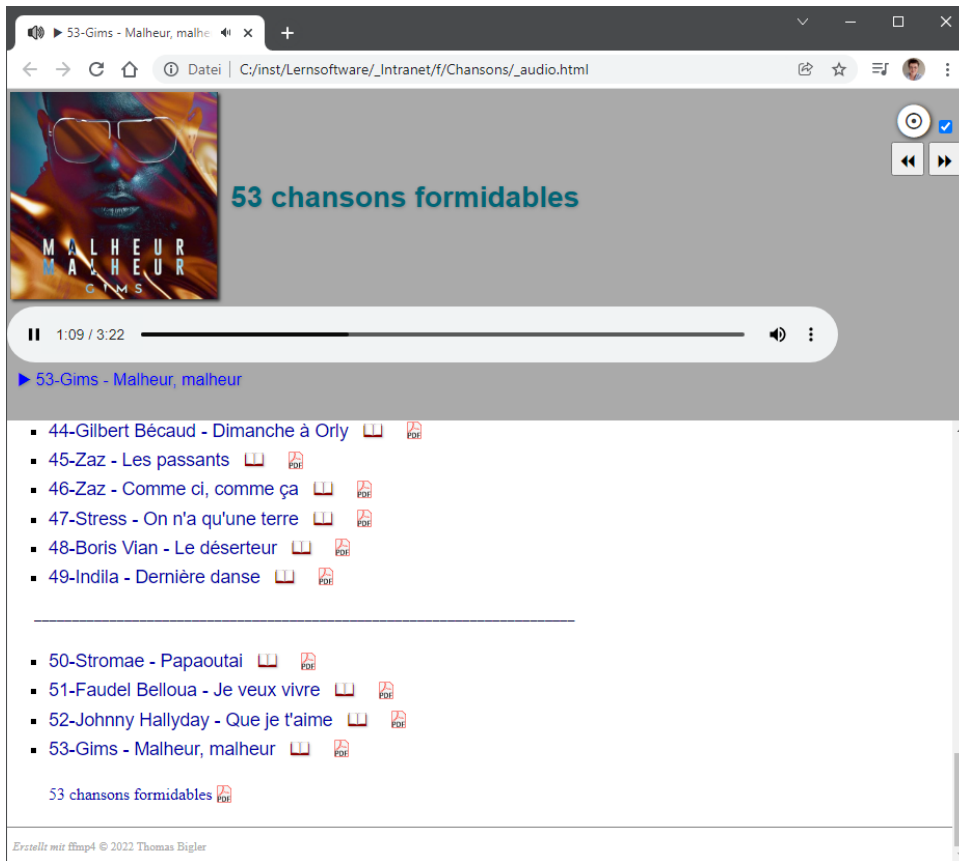
ffmp4 kann vorhandene Untertitel extrahieren: **Video: Kopie** und **Audio: Kopie** wählen – nun werden in der Quelldatei vorhandene Untertitel im Format **SRT** und **VTT** extrahiert (die VTT-Datei kann für den Media-Player verwendet werden).

Tipp:

Auf [YouTube](#) findet man viele gute Beispiele von Bildschirmaufnahmen...

Probieren geht über Studieren...


Audios präsentieren




ffmpeg4 – Audio-Medienplayer, verwendete Dateien pro Listeneintrag (Beispiel):

53-Gims - Malheur, malheur.m4a	Audiodatei, französisch (obligatorisch!)
53-Gims - Malheur, malheur.jpg	Cover
53-Gims - Malheur, malheur.txt	Legende
53-Gims - Malheur, malheur.pdf	Begleittext
folder.jpg	generisches Deckbild
folder.pdf	Informationstext, unten

Auch die Kombination von Audio mit PDF und Videos (ideal für Lehrmittel) ist möglich durch den Einsatz von Dummy-Audiodateien (leere Textdateien, umbenannt nach ***.m4a**):

- **005 - A4: Practicamos la pronunciación** [0:29]
- **006 - A5: Los números del 0 al 20** [2:08]
- **007 - D2: El pasaporte de mi avatar de español** 

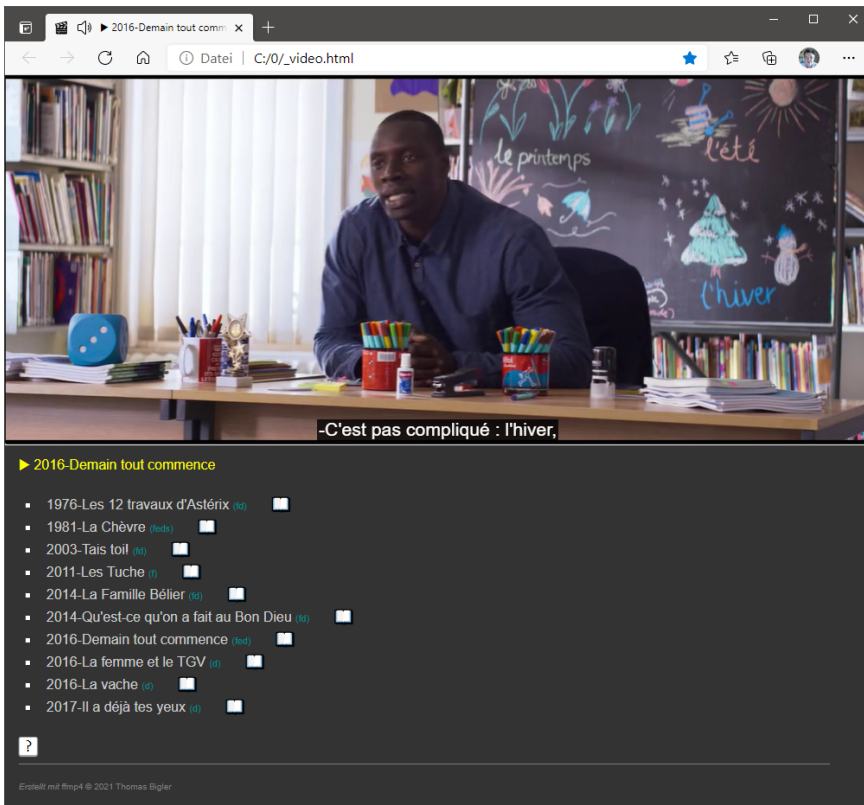
Repaso 1

- **008 - V1: ¿Cómo te llamas?** 
- **009 - A6: ¿Qué números faltan?** [0:57]

Verwendete Dateien (Beispiel):

005 - A4: Practicamos la pronunciación.m4a	Audiodatei
006 - A5: Los números del 0 al 20.m4a	Audiodatei
007 - D2: El pasaporte de mi avatar de español.m4a	Dummy-Audiodatei, (0 Bytes!)
007 - D2: El pasaporte de mi avatar de español.pdf	sichtbare PDF-Datei
008 - V1: ¿Cómo te llamas?.m4a	Dummy-Audiodatei, (0 Bytes!)
008 - V1: ¿Cómo te llamas?.mp4	sichtbare Video-Datei
009 - A6: ¿Qué números faltan?.m4a	Audiodatei

Videos präsentieren



ffmp4 – Video-Medienplayer, verwendete Dateien pro Listeneintrag (Beispiel):

2016-Demain	tout	commence.mp4	Videodatei (obligatorisch!)
2016-Demain	tout	commence.txt	Legende
2016-Demain	tout	commence.vtt	Untertitel, Vorgabe
2016-Demain	tout	commence.fr.vtt	Untertitel, französisch
2016-Demain	tout	commence.de.vtt	Untertitel, deutsch
2016-Demain	tout	commence.en.vtt	Untertitel, englisch

YouTube Videos downloaden

Im Systemmenü (Klick auf **ffmp4** im Titelbalken) befindet sich der Eintrag **YouTube Download ...**


Beim ersten Aufruf lädt diese Funktion die aktuelle Version des [Open Source](#)-Programms [yt-dlp](#) herunter, welches den Download von YouTube-Videos ermöglicht.

Dieses Hilfsprogramm wird hier abgelegt: `%USERPROFILE%\youtube-dl\youtube-dl.exe`. Da YouTube von Zeit zu Zeit seine internen Mechanismen ändert, kann es zu Problemen beim Download kommen. Durch das Löschen dieser Datei wird das Herunterladen der aktuellsten Version forciert – et voilà.

Nun kann die von **ffmp4** erstellte [Batchdatei](#) ihre Aufgabe erfüllen:

- Im Browser die Youtube-URL, z.B. <https://www.youtube.com/watch?v=JAsfAuvFvh8> mit **Ctrl** + **C** kopieren
- Die URL in YouTube-DL mit **Ctrl** + **V** einfügen
- Die **Enter**-Taste drücken und der Download des Videos beginnt:


```
Administrator: YouTube-DL
YouTube-DL -> C:\Users\Thomas\Videos\
YouTube-URL eingeben (-U = Update) : https://www.youtube.com/watch?v=CTsB-11Tzyc
```

Das heruntergeladene Video wird im *Quellordner* von  **ffmpeg4** gespeichert.

Für die höchstmögliche Videoauflösung **YouTube Download** ... mit gedrückter **Shift**-Taste starten: das Video wird ins MP4-Format konvertiert und enthält die höchste verfügbare Auflösung und ggf. auch Untertitel.

Vorsicht ist geboten bei der Suche nach ähnlichen Hilfsprogrammen im Netz: [Malware-Gefahr!](#) Die meisten dieser Angebote benutzen ohnehin auch das unübertroffene [youtube-dl!](#)

Erweiterte Anforderungen

Das sehr kleine und praktische  **ffmpeg4** deckt den Grundbedarf recht gut ab, aber manchmal möchte man mehr...

Für erweiterte Anforderungen empfehle ich folgende bewährte [Open Source](#) Programme:

- [Avidemux](#) Videoschnittsoftware, Kompression in und von den gängigsten Formaten.
- [Handbrake](#) Umfangreicher Video-Transkodierer. Erfordert vertiefte Einarbeitung.
- [MKVToolNix](#) Tools zum Bearbeiten von [Matroska](#)-Multimediadateien. Ermöglichen u.a. mehrsprachige (Audio & Untertitel) Videodateien.
- [Subtitle Edit](#) Untertitel erstellen, bearbeiten, konvertieren, es gibt nichts besseres...
- [yt-dlp](#) Videos von Streaming-Plattformen (z.B. YouTube) herunterladen. Professionelles Konsolenprogramm, erfordert Einarbeitung (eine weiterentwickelte Version von [youtube-dl](#)).

Viel Erfolg bei Download, Videokompression,
Bildschirmaufnahme und Präsentation wünscht

