FFmpeg-Workshop

Reto Kromer • AV Preservation by reto.ch

FFmpeg-Workshop
Hochschule der Künste Bern
25. Mai 2023

Benutzerschnittstellen

- Lochkarten und Matrixdrucker
- Kommandozeile
  (Englisch: command-line interface = CLI)
- grafische Benutzeroberfläche
  (Englisch: graphical user interface = GUI)
- berührungslose Schnittstellen

ASCII (1977/1986)

	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
0x	NUL	SOH	STX	ETX	EOT	ENQ	ACK	BEL	BS	HT	LF	VT	FF	CR	SO	SI
1x	DLE	DC1	DC2	DC3	DC4	NAK	SYN	ETB	CAN	EM	SUB	ESC	FS	GS	RS	US
2x	SP	!	"	#	$	%	&	'	(	)	*	+	,	-	.	/
3x	0	1	2	3	4	5	6	7	8	9	:	;	<	=	>	?
4x	@	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O
5x	P	Q	R	S	T	U	V	W	X	Y	Z	[	\	]	^	_
6x	`	a	b	c	d	e	f	g	h	i	j	k	l	m	n	o
7x	p	q	r	s	t	u	v	w	x	y	z	{	|	}	~	DEL

Changed or added in 1963 version
Changed in both 1963 version and 1965 draft

source: wikipedia.org

Unix/Linux-Befehlsstruktur

`$0` **`$1`** **`${n}`**

```
command argument_1 ... argument_n
```

Beispiele üblicher Syntaxen der Argumenten:

```
--parameter
--parameter=value
-p
-p value
```

Software

FFmpeg wird überall eingesetzt

- VLC und mpv
- Audacity und Handbrake
- QCTools und AEO-Light
- vrecord
- Google Chrome und YouTube
- «et cetera et cetera et cetera»

Die FFmpeg-Familie

Programme

- ffmpeg
- ffprobe
- ffplay

Bibliotheken

- libavcodec
- libavformat
- libavfilter
- libavutil
- libavdevice
- libswscale
- libswresample
- libpostproc

Dateiumwandlungen

ffmpeg (CLI)

➔ ffmpeg.org

FFmpeg Cookbook for Archivists

➔ avpres.net/FFmpeg/

ffmprovisr

➔ amiaopensource.github.io/ffmprovisr/

Metadaten extrahieren

MediaInfo (GUI, CLI)
➔ mediaarea.net/MediaInfo
ffprobe (CLI)
➔ ffmpeg.org

Mediaplayer

VLC (GUI)
➔ www.videolan.org/vlc/
mpv (CLI/GUI)
➔ mpv.io
ffplay (CLI)
➔ ffmpeg.org

Grafische Benutzeroberfläche

FFCommand Engine (GUI)
➔ github.com/ColorlabMD/FFCommand_Engine

FFmpeg-Befehlsstruktur

```
  $0        $1             ${n}
command argument_1 ... argument_n
```

FFmpeg-Syntax der Argumenten:

-parameter
-parameter value
-p
-p value

FFmpeg Syntax

ffmpeg
  [*global_options*]
  [*input_options_n*] **-i** *input_file_n*
  [*output_options_n*] *output_file_n*

ffprobe [*input_options*] *input_file*

ffplay [*input_options*] *input_file*

Programmbibliotheken

- demultiplexen: libavformat
- decodieren: libavcodec
- ändern: libavfilter
- codieren: libavcodec
- multiplexen: libavformat

Übungen

Einstieg

Arbeitsfolder setzen

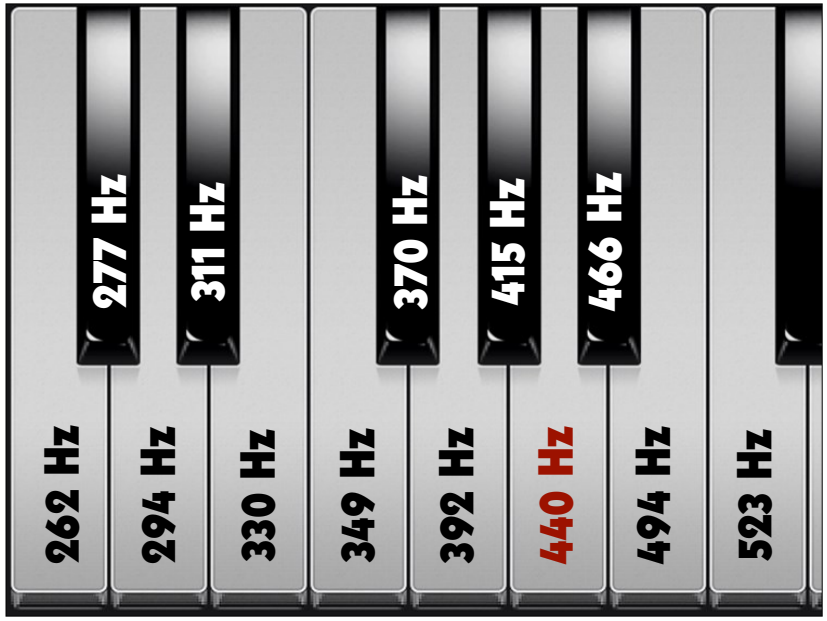
```
# Linux
# Mac
# Windows Terminal oder WSL
# Windows lokal
cd Desktop

# Windows auf OneDrive Cloud
cd OneDrive
```


Bilddatei erzeugen

```
ffmpeg
  -f lavfi -i mandelbrot
  -t 10
  -c:v rawvideo
  -pix_fmt uyvy422
  mandelbrot.avi
```


Bilddatei abspielen

```
ffplay
  mandelbrot.avi
```



Tondatei erzeugen

```
ffmpeg
  -f lavfi -i "sine=frequency=440"
  -t 10
  -c:a pcm_s16le
  -ar 48k
  -ac 2
  la.wav
```


Tondatei abspielen

```
ffplay
  la.wav
```


Dateien zusammenfügen

```
ffmpeg
  -i mandelbrot.avi
  -i la.wav
  -c:v copy
  -c:a copy
  mandela.avi
```

AV-Datei abspielen

```
ffplay
  mandela.avi
```


Metadaten extrahieren (1)

```
ffprobe
  mandela.avi
```


Container

```
ffprobe
  -show_format
  mandela.avi
```


Codec

```
ffprobe
  -show_streams
  mandela.avi
```

Container und Codec

```
ffprobe
  -show_format
  -show_streams
  mandela.avi
```


Metadaten formatieren

```
ffprobe
  -show_format
  -show_streams
  -print_format json
  mandela.avi
```


Metadaten speichern

```
ffprobe
  -show_format
  -show_streams
  -print_format json
  mandela.avi
  > mandela.txt
```


Metadaten extrahieren (2)

```
mediainfo
  mandela.avi
```

Hilfe finden

```
ffmpeg -h
ffmpeg -codecs
ffmpeg -decoders
ffmpeg -h decoder=aac
ffmpeg -encoders
ffmpeg -h encoder=libx264
ffmpeg -filters
ffmpeg -pix_fmts
```


Dateiumwandlungen

Container ändern

```
ffmpeg
  -i mandelbrot.avi
  -c copy
  mandelbrot.mov
```


MD5-Prüfsummen (1)

```
ffmpeg
  -i mandelbrot.avi
  -f framemd5
  mandelbrot_avi_framemd5.txt
```

MD5-Prüfsummen (2)

```
ffmpeg
  -i mandelbrot.mov
  -f framemd5
  mandelbrot_mov_framemd5.txt
```


Prüfsummen vergleichen

```
# Linux/Mac/Windows Terminal oder WSL
diff -s
  mandelbrot_avi_framemd5.txt
  mandelbrot_mov_framemd5.txt

# Windows
fc
  mandelbrot_avi_framemd5.txt
  mandelbrot_mov_framemd5.txt
```


Einzelbilder abspielen

```
ffplay
  -loop 0
  DUFAY_TIFF/Dufay_%06d.tif
```


Dateiumwandlungen

- von Master- nach Mezzanine- und von Mezzanine- nach Zugangsdatei
- von Master- nach Zugangsdatei

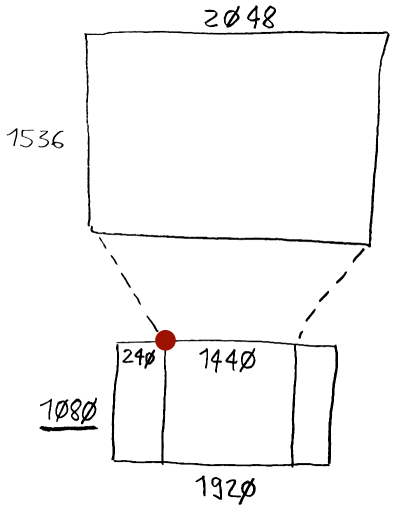
➔ Vergleiche die Qualität der Zugangsdateien

➔ Vergleiche die Qualität der Mezzaninedateien (Apple ProRes und AVID).

2K
4:3

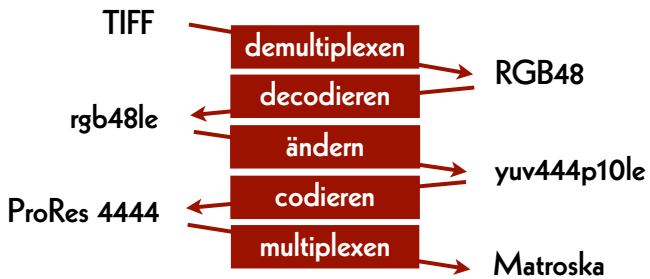
↓

HD
16:9



ProRes

Von TIFF nach ProRes



ProRes 422 und ProRes 4444

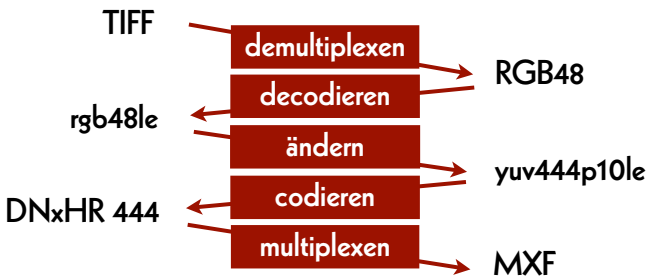
- QuickTime (.mov)
- Matroska (.mkv)
- MXF = Material eXchange Format (.mxl)

Master -> Mezzanine (1)

```
ffmpeg
  -f image2 -framerate 24
  -i DUFAY_TIFF/Dufay_%06d.tif
  -filter:v
    "scale=1440:1080:flags=lanczos,
    pad=1920:1080:240:0"
  -c:v prores_ks -profile:v 4
  Dufay_ProRes.mkv
```


AVID

Von TIFF nach DNxHR



Parameters finden

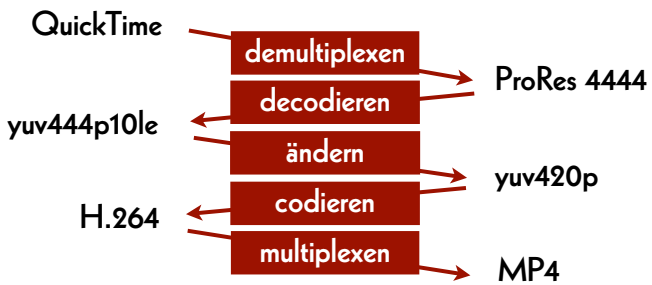
```
ffmpeg -h encoder=dnxhd

-profile:v dnxhr_lb -pix_fmt yuv422p
-profile:v dnxhr_sq -pix_fmt yuv422p
-profile:v dnxhr_hq -pix_fmt yuv422p
-profile:v dnxhr_hqx -pix_fmt yuv422p10le
-profile:v dnxhr_444 -pix_fmt yuv444p10le
-profile:v dnxhr_444 -pix_fmt gbrp10le
```

Master -> Mezzanine (2)

```
ffmpeg
  -f image2 -framerate 24
  -i DUFAY_TIFF/Dufay_%06d.tif
  -filter:v
    "scale=1440:1080:flags=lanczos,
    pad=1920:1080:240:0"
  -c:v dnxhd -profile:v dnxhr_444
  -pix_fmt yuv444p10le
  Dufay_DNxHR.mxf
```


H.264

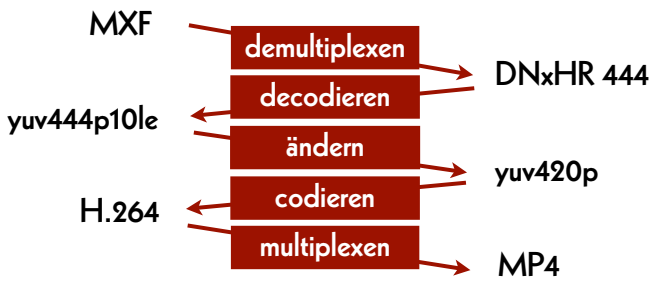
Von ProRes nach H.264



Mezzanine -> Zugang (1)

```
ffmpeg
  -i Dufay_ProRes.mkv
  -c:v libx264
  -preset veryslow
  -crf 30
  -pix_fmt yuv420p
  -movflags +faststart
  Dufay_ProRes_H264.mp4
```

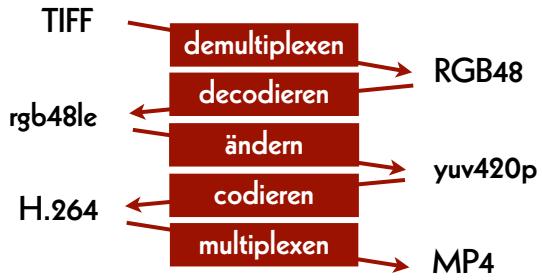
Von DNxHR nach H.264



Mezzanine -> Zugang (2)

```
ffmpeg
  -i Dufay_DNxHR.mxf
  -c:v libx264
  -preset veryslow
  -crf 30
  -pix_fmt yuv420p
  -movflags +faststart
  Dufay_DNxHR_H264.mp4
```


Von TIFF nach H.264



Master -> Zugang

```
ffmpeg
  -f image2 -framerate 24
  -i DUFAY_TIFF/Dufay_%06d.tif
  -filter:v
    "scale=1440:1080:flags=lanczos,
    pad=1920:1080:240:0"
  -c:v libx264 -preset veryslow -crf 30
  -pix_fmt yuv420p
  -movflags +faststart
  Dufay_master_H264.mp4
```

Qualitätskontrolle

Workflow

define accepted file formats

perform quality control

- checksum
- filename
- container, codec and data formats
- image and sound content

prepare archive package

store packages (e.g. onto LTO tapes)

Qualitätskontrolle

- check technical metadata
- analyse signal
- watch image and listen sound

- Differenzdatei zweier Dateien
- geteilter Bildschirm aus zwei Dateien

Quality Control Tools

container and codec

- MediaInfo, ffprobe, MediaConch
- hexdump, fq

image and sound content

- QCTools, qcli, SignalServer
- VLC, mpv, ffplay

Archival Tools

- RAWcooked
- BagIt

Geteilter Bildschirm (1)

```
ffmpeg
  -i Dufay_master_H264.mp4
  -i Dufay_ProRes_H264.mp4
  -filter_complex
    "[0] crop=iw/2:ih:0:0 [links];
    [1] crop=iw/2:ih:iw/2:0 [rechts];
    [links][rechts] hstack"
  Dufay_split_ProRes.mp4
```


Geteilter Bildschirm (2)

```
ffmpeg
  -i Dufay_master_H264.mp4
  -i Dufay_DNxHR_H264.mp4
  -filter_complex
    "[0] crop=iw/2:ih:0:0 [links];
    [1] crop=iw/2:ih:iw/2:0 [rechts];
    [links][rechts] hstack"
  Dufay_split_DNxHR.mp4
```


Geteilter Bildschirm (3)

```
ffmpeg
  -i Dufay_ProRes_H264.mp4
  -i Dufay_DNxHR_H264.mp4
  -filter_complex
    "[0] crop=iw/2:ih:0:0 [links];
    [1] crop=iw/2:ih:iw/2:0 [rechts];
    [links][rechts] hstack"
  Dufay_split_mezzanine.mp4
```

Filter testen

```
ffplay
  -vf "negate"
  Dufay_1_H264.mp4
```


Differenzdatei (1)

```
ffmpeg
  -i Dufay_master_H264.mp4
  -i Dufay_ProRes_H264.mp4
  -filter_complex
    "[1] format=yuva444p,
      lut=c3=128,
      negate [1_mit_alpha];
    [0][1_mit_alpha] overlay"
  Dufay_delta_ProRes.mp4
```


Differenzdatei (2)

```
ffmpeg
  -i Dufay_master_H264.mp4
  -i Dufay_DNxHR_H264.mp4
  -filter_complex
    "[1] format=yuva444p,
      lut=c3=128,
      negate [1_mit_alpha];
    [0][1_mit_alpha] overlay"
  Dufay_delta_DNxHR.mp4
```


Differenzdatei (3)

```
ffmpeg
  -i Dufay_ProRes_H264.mp4
  -i Dufay_DNxHR_H264.mp4
  -filter_complex
    "[1] format=yuva444p,
      lut=c3=128,
      negate [1_mit_alpha];
    [0][1_mit_alpha] overlay"
  Dufay_delta_mezzanine.mp4
```

Generate a Sound File

```
ffmpeg
  -f lavfi -i "anoisesrc=color=brown"
  -filter:a "tremolo=f=0.1:d=0.9"
  -c:a pcm_s24le
  -ar 96k
  -ac 2
  -t 60
  seashore_good.wav
```


Play the File

```
ffplay
  seashore_good.wav
```


Damage the Sound File

```
ffmpeg
  -i seashore_good.wav
  -c copy
  -bsf:a "noise=amount=-1"
  seashore_bad.wav
```


Play the Damaged File

```
ffplay
  seashore_bad.wav
```

Show Volume (good)

```
ffplay
  -f lavfi "amovie=seashore_good.wav,
  asplit [a][out1];
  [a] showvolume=c=VOLUME:
  w=1000:h=100:
  ds=lin [out0]"
```


Show Volume (bad)

```
ffplay
  -f lavfi "amovie=seashore_bad.wav,
  asplit [a][out1];
  [a] showvolume=c=VOLUME:
  w=1000:h=100:
  ds=lin [out0]"
```


Show Waves (good)

```
ffplay
  -f lavfi "amovie=seashore_good.wav,
  asplit [a][out1];
  [a] showwaves=mode=cline [out0]"
```


Show Waves (bad)

```
ffplay
  -f lavfi "amovie=seashore_bad.wav,
  asplit [a][out1];
  [a] showwaves=mode=cline [out0]"
```

Show Spectrum (good)

```
ffplay
  -f lavfi "amovie=seashore_good.wav,
  asplit [a][out1];
  [a] showspectrum=mode=separate:
  color=intensity:
  slide=1:
  scale=cbrt [out0]"
```


Show Spectrum (bad)

```
ffplay
  -f lavfi "amovie=seashore_bad.wav,
  asplit [a][out1];
  [a] showspectrum=mode=separate:
  color=intensity:
  slide=1:
  scale=cbrt [out0]"
```


Personalisieren

Schriftarten finden

```
# Linux/Mac/Windows Terminal oder WSL
ls /Library/Fonts

# Windows
dir \Windows\Fonts
```

Wasserzeichen hinzufügen

```
ffmpeg
  -i Dufay_master_H264.mp4
  -filter:v
    "drawtext=text='watermark':
    fontfile='/Library/Fonts/Arial.ttf':
    fontsize=35:
    fontcolor=white:
    alpha=0.25:
    x=(w-text_w)/2:y=(h-text_h)/2"
  watermark.mp4
```


Timecode hinzufügen

```
ffmpeg
  -i Dufay_master_H264.mp4
  -filter:v
    "drawtext=timecode='01\:00\:30\:00':
    rate=25:
    fontfile='/Library/Fonts/Arial.ttf':
    fontsize=35:
    fontcolor=white:
    x=(w-text_w)/2:y=h/1.2"
  timecode.mp4
```


Logo hinzufügen

```
ffmpeg
  -i Dufay_master_H264.mp4
  -i logo.png
  -filter_complex
    "overlay=10:main_h-overlay_h-10"
  mit_logo.mp4
```


AV Preservation by

reto.ch

Sandrainstrasse 3
3007 Bern
Switzerland

Web: reto.ch
Email: info@reto.ch