```
# 1.  Check if the software is installed and working
#     ----------------------------------------------
#
#     ffmpeg transforms AV files from one format into another
```

ffmpeg -version

```
#     ffplay is an AV file player
```

ffplay -version

```
#     ffprobe is an AV file prober
```

ffprobe -version

```
# 2.  Extract the technical metadata from an AV file
#     ----------------------------------------------
#
#     Parameters:
#     -show_format   shows the container's metadata
#     -show_streams  shows the codec's metadata
#     -print_format  choses the output format
#
#     NOTE: You can drag & drop the first file into the terminal window.
#
#     See also:
#     • https://avpres.net/FFmpeg/probe_json.html
#     • http://amiaopensource.github.io/ffmprovisr/#pull_specs
```

ffprobe DUFAY_TIFF/Dufay_000001.tif

ffprobe -show_format DUFAY_TIFF/Dufay_000001.tif

ffprobe -show_streams DUFAY_TIFF/Dufay_000001.tif

ffprobe -show_format -show_streams -print_format flat DUFAY_TIFF/Dufay_000001.tif

ffprobe -show_format -show_streams -print_format json DUFAY_TIFF/Dufay_000001.tif

ffprobe -show_format -show_streams -print_format xml DUFAY_TIFF/Dufay_000001.tif

```
# 3.  Play an AV file
#     ---------------
#
#     Parameters:
#     -framerate 5    a speed of 5 fps is set
#                     NOTE: this parameter must be before the input file
#     [none]          path, name and extension of the input files
#                     The regex %06d matches six digits long numbers,
#                     possibly with leading zeroes. This allows to read in
#                     ascending order, one image after the other, the full
#                     sequence inside one folder.
#                     The command must of course match the naming convention
#                     actually used.
#
#     See aldo:
#     • https://avpres.net/FFmpeg/play_sq.html
#     • http://amiaopensource.github.io/ffmprovisr/#play_im_seq
```

ffplay DUFAY_TIFF/Dufay_%06d.tif

ffplay -framerate 5 DUFAY_TIFF/Dufay_%06d.tif

```
# 4.  Transform the preservation master (e.g. DPX or TIFF) into a mezzanine
#     format (e.g. Apple ProRes 422 HQ or Avid DNxHD 175x)
#     ----------------------------------------------------------------------
#
#     Parameters:
#     -f image2    forces the image file de-muxer for single image files
#                  NOTE: this parameter must be before the input file
#     -i           path, name and extension of the input files
#     -c:v         we chose the ProRes codec
#     -profile:v   the video profile 3 is for HQ
#     -filter:v    we filter the video stream:
#                  • scaling to the correct size
#                    [we use the Lanczos scaling algorithm which is slower
#                    but better than the default bilinear algorithm]
#                  • padding the 4:3 format into the HD format with
#                    pillar-box
#     -an          no audio
#
#     See also:
#     • https://avpres.net/FFmpeg/sq_ProRes.html
#     • http://amiaopensource.github.io/ffmprovisr/#to_prores
```

ffmpeg -f image2 -framerate 24 -i DUFAY_TIFF/Dufay_%06d.tif -c:v prores -profile:v 3 -filter:v "scale=1440:1080:flags=lanczos,pad=1920:1080:240:0" -an Dufay_ProRes422HQ.mov

```
# 5.  Transform the mezzanine format (e.g. Apple ProRes 422 HQ or Avid
#     DNxHD 175x) into an access file (e.g. MP4/H.264)
#     ------------------------------------------------------------
#
#     Parameters:
#     -c:v      we chose the libx264 codec
#     -preset   we chose "veryslow" which gives the best result
#     -qp       a quality parameter of 18 means "visually lossless"
#     -pix_fmt  we chose "yuv420p" (YUV 4:2:0 planar) for best
#               compatibility
#
#     See also:
#     • https://avpres.net/FFmpeg/im_H264.html
#     • http://amiaopensource.github.io/ffmprovisr/#transcode_h264
```

ffmpeg -i Dufay_ProRes422HQ.mov -c:v libx264 -preset veryslow -qp 18 -pix_fmt yuv420p -an Dufay_H264.mp4

```
# 6.  Not used during this workshop
#     -----------------------------
```

ffmpeg -f lavfi -i mandelbrot -t 10 -pix_fmt yuv420p -an mandelbrot.mov

ffmpeg -i mandelbrot.mov -c:v copy mandelbrot.avi

ffmpeg -i mandelbrot.mov -f framemd5 -an mandelbrot_mov_md5.txt

ffmpeg -i mandelbrot.avi -f framemd5 -an mandelbrot_avi_md5.txt

ffmpeg -f lavfi -i aevalsrc="sin(440*2*PI*t)" -t 10 A.wav

ffmpeg -i mandelbrot.mov -i A.wav mandelaaa.mov