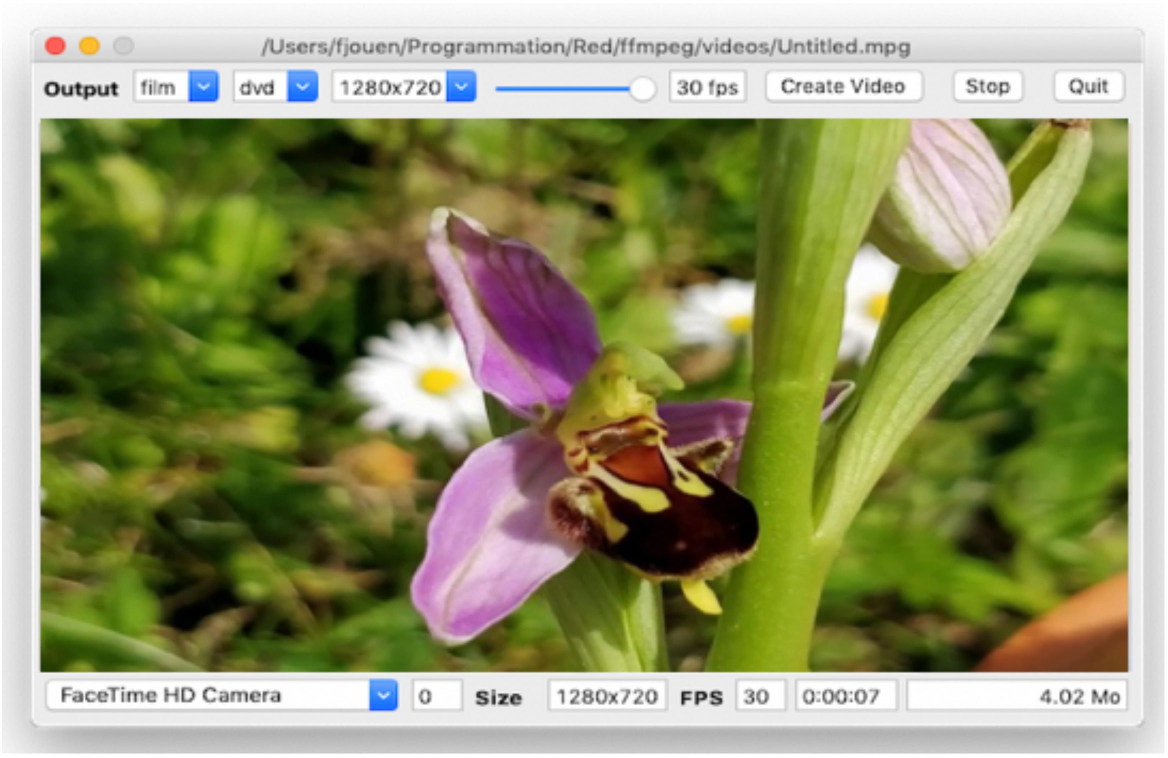
redCV and FFmpeg: Video and audio capture

FFmpeg is a fabulous command-line framework for multimedia processing. Among variety of features, FFmpeg can capture video and audio from your computer's camera and microphone. Since Red language does support video capture, it was really easy to connect both programs and realize a nice Red video recorder. Red is used to display camera images on screen and, FFmpeg to record movie.



Output film dvd 1280x720 30 fps

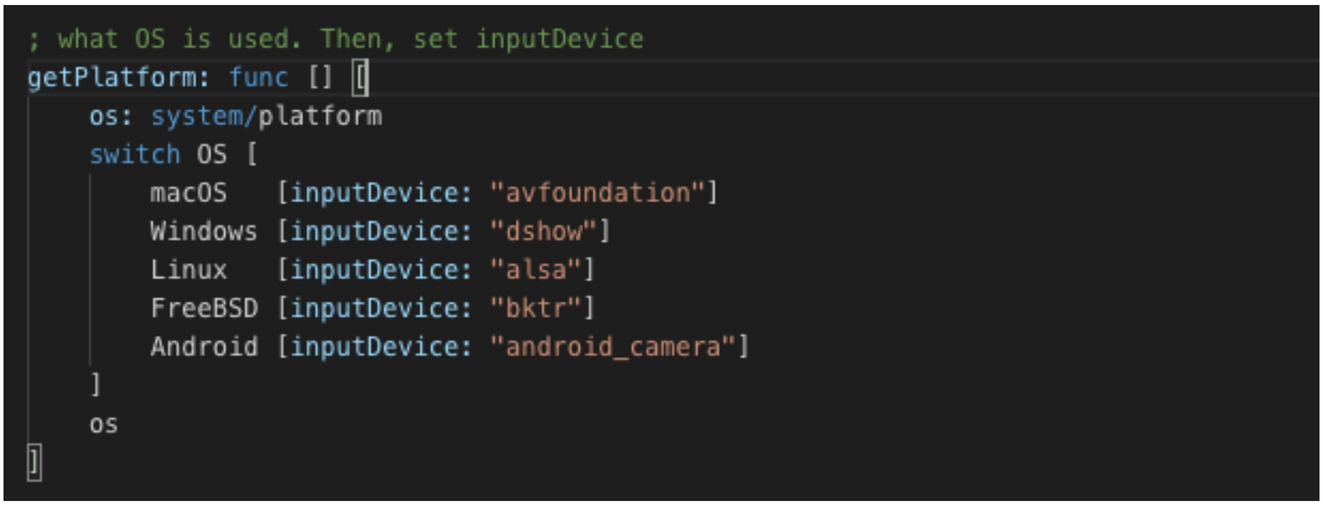
FaceTime HD Camera 0 Size 1280x720 FPS 30 0:00:07 4.02 Mo

Wiith this Red code, you can select video and audio inputs, change the quality and size of the recorded video. You can also control the frequency of recorded frames (FPS).

Supported video files for recording are *mpg, mp4, mkv, avi, wmv, and mov*.

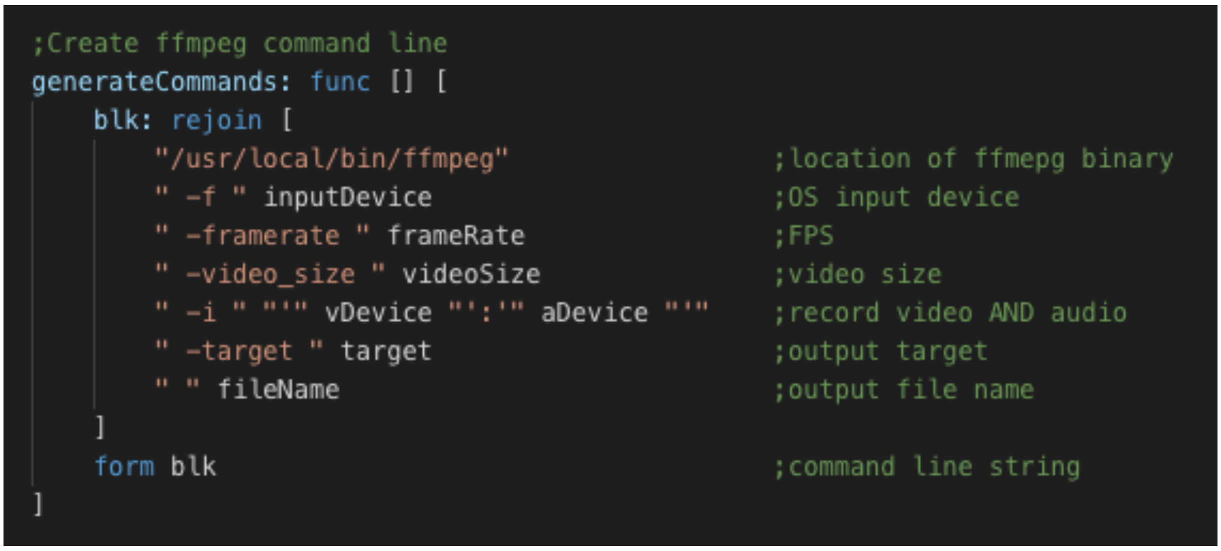
Before we start, you must have Red language (http://www.red-lang.org) and FFmpeg (https://www.ffmpeg.org/) installed on your computer and you must know the path of the FFMPEG binary such as **/usr/local/bin/ffmpeg** for Mac or Linux.

As Red, FFmpeg is cross-platform and can be used vith various operating systems. The Red *getPlateform* function is called to select the running OS and then to use the correct FFmpeg input device.



```
; what OS is used. Then, set inputDevice
getPlatform: func [] [
    os: system/platform
    switch OS [
        macOS   [inputDevice: "avfoundation"]
        Windows [inputDevice: "dshow"]
        Linux   [inputDevice: "alsa"]
        FreeBSD [inputDevice: "bktr"]
        Android [inputDevice: "android_camera"]
    ]
    os
]
```

Then, the second operation is to generate the command-line that will be passed as parameter to FFmpeg binary. This is done by Red function *generateCommands*.

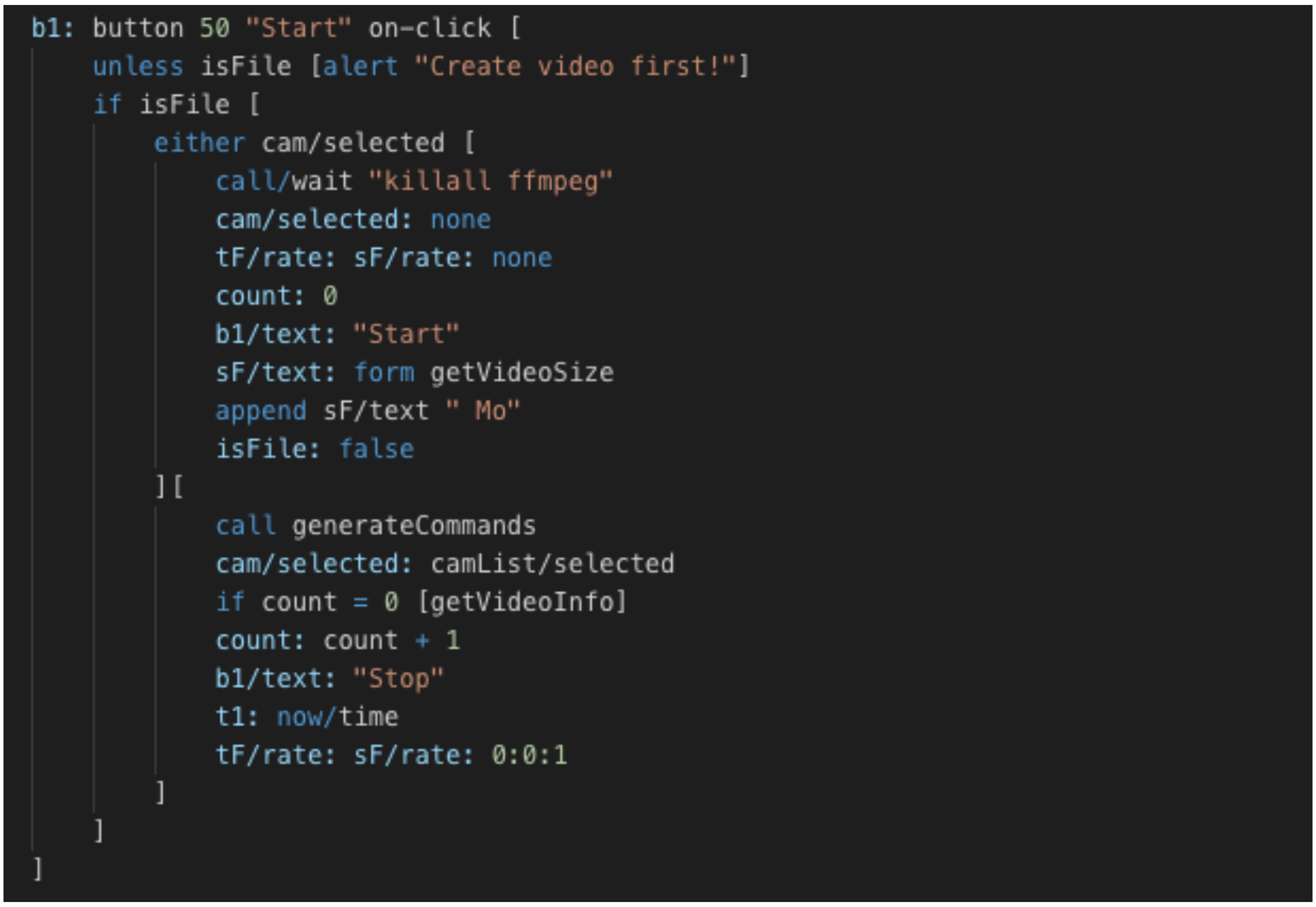


```
;Create ffmpeg command line
generateCommands: func [] [
    blk: rejoin [
        "/usr/local/bin/ffmpeg"                        ;location of ffmepg binary
        " -f " inputDevice                             ;OS input device
        " -framerate " frameRate                       ;FPS
        " -video_size " videoSize                      ;video size
        " -i " "'" vDevice "':'" aDevice "'"           ;record video AND audio
        " -target " target                             ;output target
        " " fileName                                   ;output file name
    ]
    form blk                                           ;command line string
]
```

In the code above, a few of FFmpeg options and Red words are used:

1. **-f inputDevice**: to use the OS device for grabbing video (on macOS, we use the avfoundation device).
2. **-framerate frameRate**: the FPS (1..30) for recording.
3. **-video_size videoSize**: required video size (a pair WxH, on my MacBook Pro: "1280x720" or "640x480").
4. **-i vDevice:aDevice**: the video and audio device used for recording. By default, vDevice = 0 corresponds to the first camera found on computer (e.g. Apple FaceTime Camera on macOS), and aDevice = 0 to computer microphone.
5. -**target target**: this is a combination of 2 values for determining the quality of the video (e.g film_dvd).
6. lastly the **fileName** is provided to store the video.

When FFmpeg command-line is generated, we just need Red call function to start or stop the movie grabbing.



```
b1: button 50 "Start" on-click [
    unless isFile [alert "Create video first!"]
    if isFile [
        either cam/selected [
            call/wait "killall ffmpeg"
            cam/selected: none
            tF/rate: sF/rate: none
            count: 0
            b1/text: "Start"
            sF/text: form getVideoSize
            append sF/text " Mo"
            isFile: false
        ][
            call generateCommands
            cam/selected: camList/selected
            if count = 0 [getVideoInfo]
            count: count + 1
            b1/text: "Stop"
            t1: now/time
            tF/rate: sF/rate: 0:0:1
        ]
    ]
]
```

In less than 150 lines of code, we get a very efficient movie grabber which records **both video and audio** channels.

You'll find the code here: https://github.com/ldci/ffmpeg/camera.red. Enjoy:)