F F M P E G

FROM ZERO TO HERO

NICK FERRANDO

FFMPEG
From Zero to Hero

By Nick Ferrando

FFmpeg is a trademark of Fabrice Bellard, originator of the FFmpeg Project.
Adobe Creative Cloud is a trademark of Adobe, Inc.
Apple, MacOS, OS X and Final Cut Pro X are trademarks of Apple, Inc.
Avid Media Composer is a trademark of Avid, Inc.
Bento4 is a trademark of Axiomatic Systems, LLC
ImageMagick is a trademark of ImageMagick Studio, LLC
Linux is a registered trademark of Linus Torvalds
Remove.bg and unscreen.com are trademark of Kaleido AI, GmbH.
Sublime Text is a trademark of Sublime HQ Pty Ltd.
Ubuntu is a registered trademark of Canonical, Ltd.
Windows is a registered trademark of Microsoft Corp., Ltd.

Cover Illustration by Tarik Vision, Licensed by Getty Images.

DISCLAIMER

This book contains copyrighted material, such as book extracts or blog extracts, graphics, logos and pictures the use of which has not always been specifically authorized by the original copyright owner. I'm making such material available in my effort to teach and advance understanding of computer technology, graphics, video editing, video compression technology, software development and computer programming. I believe that this constitutes a fair use of any such copyrighted material as provided for in Section 107 of the US Copyright Law.

All trademarks, names and services described in this book are used for educational purposes and are property of their respective owners.

www.ffmpegfromzerotohero.com

Index

Acknowledgments

Thank you for reading this book.
This is actually my very first technical book on the subject and i do hope you will find it useful for your audio and video content production needings.
It has been written with the goal to provide a quick and effective way to understand and use FFMPEG along with many other great open source technologies used to create, edit and process audio, video and pictures at scale.
This book will provide several practical formulas and their syntax explanation.
FFMPEG won't be the only piece of software discussed in this book: there are quite few tools that works in conjunctions with FFMPEG and some formulas and tools that you will discover, or you may re-discover, that will assist you for your professional needs.
By writing this book I wanted to share my 20+ years of experience with content production, and particularly my last years of experience with video production and automation.

A special acknowledgment and a special thank you goes to my dear friend *Andy Lombardi*:
Andy: you are my digital guru and a true friend whose knowledge and humanity are a continuos ispiration for me.

Of course this book won't exist without the genius mind of *Fabrice Bellard*, the creator of FFMPEG and all the FFMPEG active developers around the world.

A special dedication goes to *Leonardo Chiariglione*, *Hiroshi Yasuda*, *Federico Faggin*, *Dennis Ritchie*, *Ken Thompson*, Stephen Bourne, Steve Wozniak, Steve Jobs, *Richard Stallman* and *Linus Torvalds*.

With infinite admiration to the entire Ferrando family.

What is FFMPEG

Ffmpeg is a very fast video and audio converter that can also grab from a live audio/video source. It also reads from an arbitrary number of input "files" which can be your own computer files, pipes[1], network streams or URLs, grabbing devices, etc.[2]

If you ever wondered how the developers of **YouTube** or **Vimeo** cope with billions of video uploads or how **Netflix** processes its catalogue at scale or, again, if you want to discover how to create and develop your own video platform, you may want to know more about FFMPEG.

This acronym stands for "**F**ast-**F**orward-**M**oving-**P**icture-**E**xpert **G**roup".

The Moving Picture Experts Group, **MPEG**, is a working group of authorities that was formed in 1988 by the standard organization **ISO,** The International Organization for Standardization, and **IEC**, the International

[1] pipe is a technique for passing information from one program process to another.

[2] https://ffmpeg.org/ffmpeg.html#toc-Description

Electrotechnical Commission, to set standards for audio and video compression and transmission.

Since its establishment by Leonardo Chiariglione and Hiroshi Yasuda, the **M**oving **P**ictures **E**xperts **G**roup has made an indelible mark on the transition from analog to digital video[3].

FFMPEG is by definition a framework, which can be defined as a platform or a structure for developing software applications. FFMPEG is able to process pretty much anything that humans have created in the last 30 years or so, in terms of audio, video, data and pictures.

It supports the most obscure old formats up to the cutting edge, no matter if they were designed by some standards committee, the community or a corporation[4].

In the last 10 years the content creation has seen an incredible evolution and expansion: if you are a content creator yourself, you will be familiar with tons of the on-line tools, Apps, or subcription based platforms such as the

---

[3] https://www.streamingmedia.com/Articles/Editorial/Featured-Articles/MPEG-What-Happened-141678.aspx

[4] http://ffmpeg.org/about.html

Adobe Creative Cloud or cutting-edge editing softwares such as FinalCut Pro or Avid Media Composer.

FFMPEG is not a substitute of those softwares, but at the same time it can perform many of their tasks in a smarter, faster and costless way.

Intended Audience

This book is designed to address anyone who is just above the “raw beginner” level. This book will explain some basic process such as entering commands and execute simple code instructions using a **C**ommand-**L**ine-**I**nterface, or "CLI", instead of using high resource-intensive **G**raphical **U**ser **I**nterfaces, or "GUI".

You may review some basic definitions and concepts, or skip directly to the working Formulas, as you'll prefer.

Whether you are at the very beginning or an experienced developer, you will find several effective ways to execute many tasks for your audio/video/streaming needings.
A great deal of the technology discussed in this book is an evolution of discoveries in the field of computer science mainly developed in the early 1970 by Dennis Ritchie and

Ken Thompson: a lot of technology developed back then is still with us today and it will continue to be for a long time.

All the software discussed in this book is mostly free and open-source and is developed by extremely talented developers around the world.

Two Google engineers, for example, have been amongst the major contributors of the FFMPEG project[5].

A chapter of this book is entirely dedicated for the basic definitions of most of the technical terms used in this text.

Tested Platforms

All the instructions and Formulas described in this book have been successfully tested on a MacBook Pro with MacOS X Catalina 10.15.6, on Ubuntu 18.04 and 20.04.

For Windows users: while there is a way to install FFMPEG as a standalone executable .exe, i suggest you to install the BASH Shell for Windows by following the step-by-step guide available here:

`https://docs.microsoft.com/en-us/windows/wsl/install-win10`

[5] FFMPEG and a thousand fixes - Google Blog: https://security.googleblog.com/2014/01/ffmpeg-and-thousand-fixes.html

Basic Definitions

As mentioned before, in order to use all the programs and tools described in this book you will need to use a "Shell" and more specifically the BASH shell.

SHELL: Is a UNIX term for a user interface to the system: something that lets you communicate with the computer via the keyboard and the display thru direct instructions (Command Lines) rather than with a mouse and graphics and buttons. The shell's job, then, is to translate the user's command lines into operating system instructions[6].

BASH: Bash is the shell, or command language interpreter, for Unix-like systems.
The name is an acronym for the '**B**ourne-**A**gain **SH**ell', a pun on Stephen Bourne, the author of the direct ancestor of the current Unix shell "**sh**", which appeared in the 7th Edition Bell Labs Research version of Unix, in 1979.

6 Newham, Cameron. Learning the bash Shell (In a Nutshell) (O'Reilly)

ENCODE: The process to compress a file so to enable a faster transmission of data.

DECODE: The function of a program or a device that translates encoded data into its original format.

CODEC: A codec is the combination of two words en**CO**der and **DEC**oder. An encoder compress a source file with a particular algorithm: then a decoder can decompress and reproduce the resulting file.
Common examples of video codecs are: MPEG-1, MPEG-2, H.264 (aka AVC), H.265 (aka HEVC), H.266 (aka VVC), VP8, VP9, AV1, or audio codecs such as Mp3, AAC, Opus, Ogg Vorbis, HE-AAC, Dolby Digital, FLAC, ALAC.

BITRATE: Bitrate or data rate is the amount of data per second in the encoded video file, usually expressed in kilobits per second (kbps) or megabits per second (Mbps). The bitrate measurement is also applied to audio files.
An Mp3 file, for example, can reach a maximum bitrate of 320 kilobit per second, while a standard CD (non-compressed) audio track can have up to 1.411 kilobit per

second. A typical compressed h264 video in Full-HD has a bitrate in the range of 3.000 - 6.000 kbps, while a 4k video can reach a bitrate value up to 51.000 kbps.
A non-compressed video format, such as the Apple ProRes format in 4K resolution, can reach a bitrate of 253.900 kbps and higher.

CONTAINER: Like a box that contains important objects, containers exist to allow multiple data streams, such as video, audio, subtitles and other data, to be embedded into a single file. Amongst popular containers there are:
MP4 (.mp4), MKV (.mkv), WEBM (.webm), MOV (.mov), MXF (.mxf), ASF (.asf), MPEG Transport Stream (.ts), CAF (Core Audio Format, .caf), WEBP (.webp).

MUX: This is the process of taking encoded data in the form of 'packets' and write it into files, in a specific container format.

DEMUX: The process of reading a media file and split it into chunks of data.

TRANSMUXING: Also referred to as "repackaging" or "packetizing", is a process in which audio and video files are repackaged into different delivery formats without changing the original file content.

TRANSCODING: The process of converting a media file or object from one format to another.

RESOLUTION: Resolution defines the number of pixels (dots) that make up the picture on your screen.

For any given screen size the more dots in the picture, the higher the resolution and the higher the overall quality of the picture. TV resolution is often stated as the number of pixels or dots contained vertically in the picture.
Each of these resolutions also has a number and a name associated with it. For example: 480 is associated to SD (Standard Definition). 720 and 1080 are associated to HD (High-Definition), 2160 is associated to UHD (Ultra-High-Definition) and finally 4320 is associated to 8K UHD.

ASPECT RATIO: This is the ratio (relation) of width to height of the TV screen. Certain aspect ratios are designed to handle certain resolutions without any stretching or distortion of the picture and without any blank space around the picture. Common aspect ratios are the "4:3 aspect ratio" meaning that for every 4 inches of width in an image, you will have 3 inches of height. Another popular aspect ratio is 16:9, meaning thath for every 16 inches of width in an image, you will have 9 inches of height.

INTERLACED FORMAT: It's a technique invented in the 1920s to display a full picture by dividing it into 2 different set of lines: the even and the odd lines.
The even lines are called "even field", while the odd lines are called the "odd field".

The even lines are displayed on the screen, then the odd lines are displayed on the screen, each one every 1/60th of a second: both of these, even and odd fields, make up one video frame. This is one of the earliest video compression methods[7].

---

[7] https://becg.org.uk/2018/12/16/interlacing-the-hidden-story-of-1920s-video-compression-technology/

PROGRESSIVE: Progressive format refers to video that displays both the even and odd lines, meaning the entire video frame, at the same time.

LETTER BOX: Letterboxing is the practice of transferring film shot in a widescreen aspect ratio to standard-width video formats while preserving the content's original aspect ratio. The resulting videographic image has black bars above and below it. LBX or LTBX are the identifying abbreviations for films and images so formatted.

VOD/SVOD: Acronym for **V**ideo **O**n **D**emand / **S**ubscription-based **V**ideo **O**n **D**emand.

OTT: Abbreviation for "Over-the-top". A video streaming service offered directly to the public thru an internet connection, rather than thru an antenna, cable or satellite.

STREAMING: The process of delivering media thru small chunks of compress data and sent to a requesting device.

RTMP: Real-Time-Messaging-Protocol. A proprietary protocol originally developed by Macromedia, today Adobe.

HLS: HTTP-**L**ive-**S**treaming Protocol. Created by Apple for delivering video destined for Apple devices.

DASH: Dynamic **A**daptive **S**treaming over **H**TTP.
This protocol can be defined as the open-source HLS.

M3U8: A standard text file encoded in UTF-8[8] format and organized as a playlist of items with their location, to be reproduced in a particular sequence.

AV1: Stands for AOMedia Video 1, which is an acronym for Alliance for Open Media Video 1, a codec developed in joint-venture by Amazon, Cisco, Google, Intel, Microsoft, Mozilla, Neflix, ARM, NVidia, Apple and AMD.

---

8 UTF-8 stands for Universal Text Format 8

BATCH PROCESSING: The act of processing a group of catalogued files.

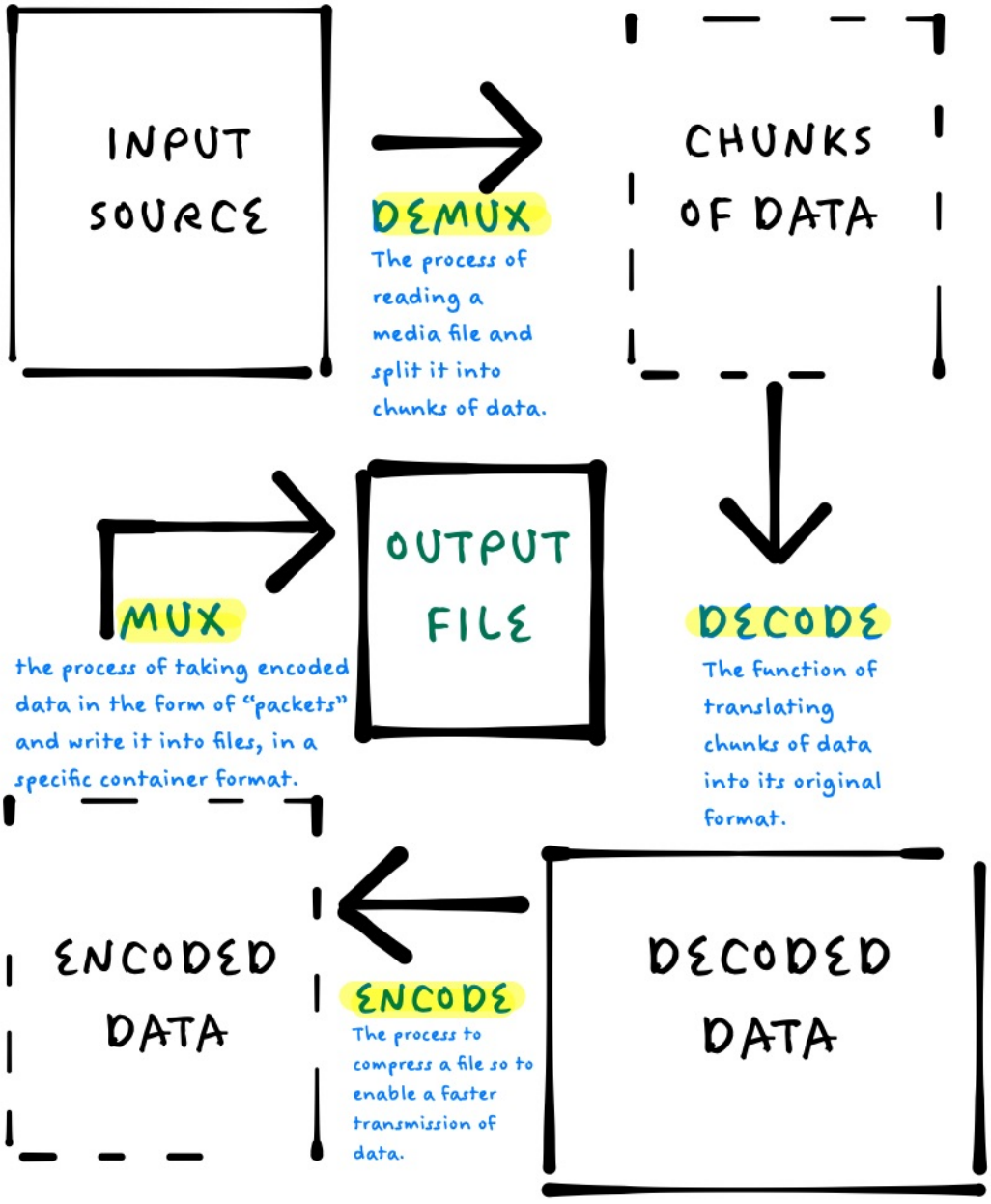
HDR: Acronym for High-Dynamic-Range. It is about recreating image realism from camera through postproduction to distribution and display[9].

h264: A standard of video compression defined by the Motion Expert Picture Group (MPEG) and the International Telecommunication Union (ITU). It is also referred to h264/AVC (Advanced Video Coding).

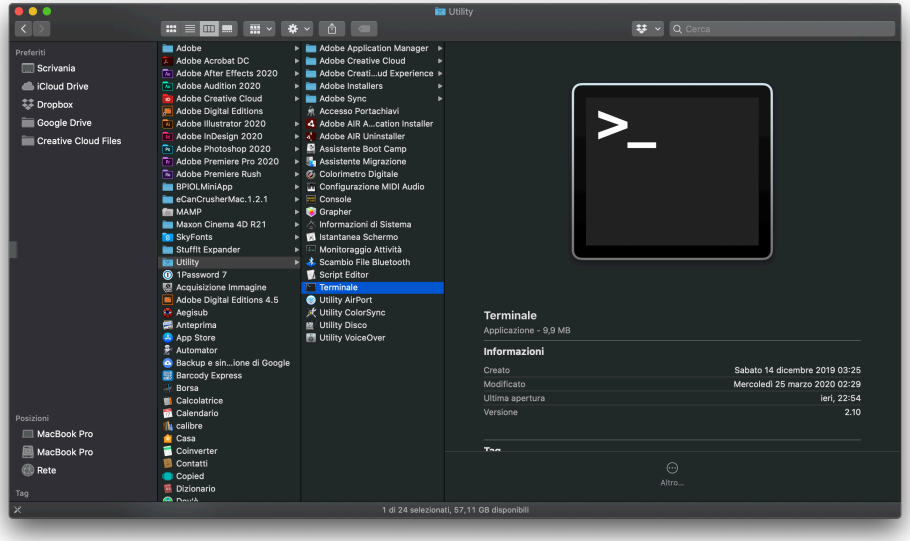
x264: a free software library and application for encoding video streams into the H.264/MPEG-4 AVC compression format.

[9] https://aws.amazon.com/media/tech/what-high-dynamic-range-hdr-video/#:~:text=High%20dynamic%20range%20(HDR)%20video%20technology%20is%20the%20next%20great,postproduction%20to%20distribution%20and%20display.

Basic FFMPEG Workflow



How to Install FFMPEG



MacOS X contains a great app called "Terminal" which is inside the Utility Folder.

The Terminal application on MacOS X it's a standard shell with a pre-installed BASH version 3.

Please note: the latest MacOS X version use an extended version of BASH named ZSH (aka "Zed Shell"). In order to use all the tools and commands described in this book i suggest you to switch from **ZSH** to **BASH**.

To do so, just type the word *bash* as soon as your Terminal will open, and press Enter. This will enable the BASH shell immediately.

First thing to do with Terminal: choose or locate your default working directory.

By default Terminal will process your commands on the current USER directory path (aka your Home Directory). You will find lots of tilde symbol (~) in this book: that tilde symbol means "your home directory".

To access your "Desktop" directory on MacOS X, for example, you can type the following command on Terminal:

```
cd ~/Desktop
```

The above command means "**C**hange the **D**irectory where i am located, to the user Desktop". You just want to make sure that everything you will output from your Terminal will be easily located on your computer, once the processes are done.

For example: you might want to process a large batch of videos. In this case it will a good idea to can create a folder on your desktop and call it "processed"

To create this folder you will type:

```
mkdir ~/Desktop/processed
```

To access this folder on your Desktop, within Terminal, you will type:

```
cd ~/Desktop/processed
```

While there are several ways to install FFMPEG on MacOS X or on Ubuntu, i suggest you to download and install one specific program called "HomeBrew" thru the Terminal Application.

Just open the Terminal Application on MacOS X and then paste the following code:

```
/bin/bash -c "$(curl -fsSL https://
raw.githubusercontent.com/Homebrew/install/
master/install.sh)"
```

The above mentioned code is a BASH command that basically says: "*Please BASH run the command within the quotes so to download and install HomeBrew from the HomeBrew's Website onto my computer*".

For the sake of simplicity, the above command won't be analyzed word by word. A complete techcnical explanation of the command can be founded on the **Additional Notes** of this book and/or by visiting the Bash Reference Manual available at the GNU.org website.

Install a Basic FFMPEG version thru HomeBrew

After installing HomeBrew it's time to install FFMPEG. At the time of writing this book the latest version is 4.3.1.

Now type the following code:

```
brew install homebrew-ffmpeg/ffmpeg/ffmpeg
```

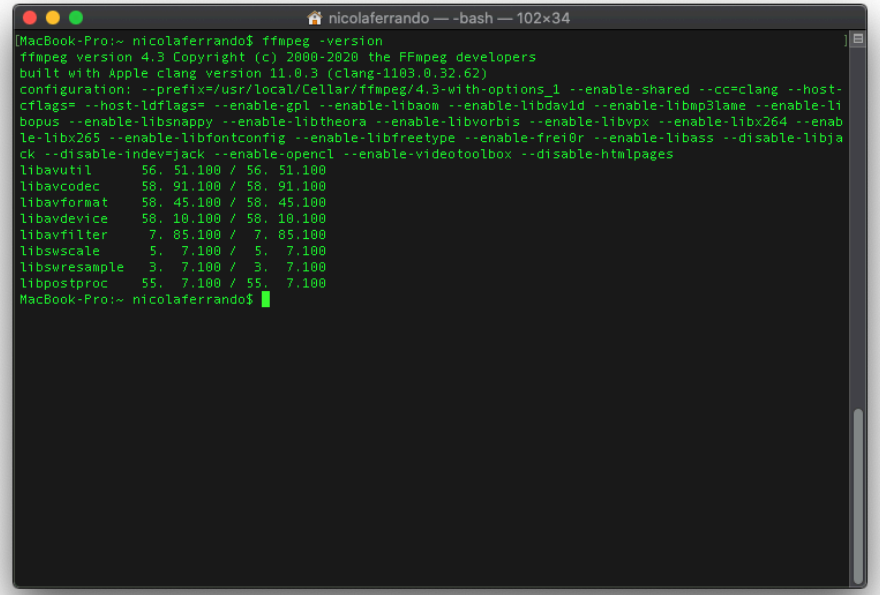
This might take 5 minutes or more.

After completion of the installation process, you might want to check if everything is fine with your installed FFMPEG.

To do so, within your Terminal window, just type the following code:

```
ffmpeg -version
```

Your Terminal will output something like this:



If you are seeing a similar screen, after running the command above, that means that the installation process of FFMPEG on MacOS X has completed.

Optional: Custom Installation of FFMPEG on MacOS X with Options

You might want to use an FFMPEG version that has all the protocols and libraries needed for a particular task.

For example, you might need to use a **Decklink** acquisition card from BlackMagic Design.
In this case you can check the options available on Brew by typing the following command:

```
brew info homebrew-ffmpeg/ffmpeg/ffmpeg
```

This will print out all the supported options.
In order to install FFMPEG with the desired options, you will need to type the command along with the desired option to install. In the example of DeckLink support, you will have to install the Blackmagic Desktop Video and Driver libraries and then type the following command:

```
brew install homebrew-ffmpeg/ffmpeg/ffmpeg
--with-decklink
```

As mentioned before FFMPEG can be installed in many ways, and with some options that can process many formats and standards created in the last 30 years. Amongst these formats there are patented formats, such as the common Mp3 format, which is a registered patent of the **Fraunhofer Institute**, which deals with licenses and authorized uses for commercial purposes.

For example: if you want to use a proprietary codec such as the AAC into an App or a commercial software, and embed FFMPEG as a component of your App, you might want to study **this legal section of the FFMPEG website**.

As per the Blackmagic Design option above described, if you need to use FFMPEG on a specific configuration or output a special format you may want to install a custom version of FFMPEG.

Let's take an example: the AAC audio compression codec.

This audio codec can be used freely by calling the native AAC encoder option with the command **`-c:a aac`** (which means "Codec for Audio: AAC"). But you might need to use a special version of the AAC codec, such as the HE-AAC (High Efficiency Advance Audio Coding).

To use this patented format, which is yet another Fraunhofer Institute Patent, you will need to install FFMPEG with the **`libfdk_aac`** library.

To enable this option you will have to type:

```
brew install homebrew-ffmpeg/ffmpeg/ffmpeg
--with-fdk-aac
```

From a quality standpoint the **`libfdk_aac`** option will produce superior results from the native FFMPEG's AAC encoder. You might want to investigate further this point, by reading this **FFMPEG Guideline** for high-quality lossy audio encoding.

On MacOS X and with HomeBrew installed, you can check all the available install options of FFMPEG, by typing on your Terminal the following command:

```
brew options homebrew-ffmpeg/ffmpeg/ffmpeg
```

Installing FFMPEG 3.X on Linux (Ubuntu)

On Ubuntu releases you can open the Terminal Application which is already installed on the operating system.

To install FFMPEG, open Terminal and type the following set of 2 command:

```
sudo apt update&&sudo apt install ffmpeg
```

Please keep in mind that this Ubuntu version of FFMPEG might not be the same version installed with HomeBrew on a Mac OS system.

sudo = "Super User Do!". This means you are asking the system to perform actions as the main administrator, with full privileges. You might be asked to enter an Administrator Password before proceeding.

apt = Advance Package Tool. Is the most efficient and preferred way of managing software from the command line for Debian and Debian based Linux distributions like Ubuntu.

update = This will update all the software. This is the step that actually retrieves information about what packages can be installed on your Ubuntu system, including what updates to currently installed packages are available.

&& = this is used to chain commands together, such that the next command is run *if and only if* the preceding command exited without errors

apt install ffmpeg = this is the command used to install the package FFMPEG, available on the Advance Package Tool.

Installing FFMPEG 4.x on Ubuntu

```
sudo apt install snapd&&sudo snap install ffmpeg
```

Please note that this version of FFMPEG will install also the "non-free" option and licensed codecs, such as the **`libfkd_aac`**. Please refer to the above Custom Installation section for legal terms and conditions that might apply in your specific development case.

Installing FFMPEG 4.x on Ubuntu with all the Bells and Whistles

FFMPEG can be installed in many ways. As discussed earlier you might use a Mac, therefore you might want to install FFMPEG thru HomeBrew or by following the official instructions on how to compile FFMPEG with custom dependancies, available here:

https://trac.ffmpeg.org/wiki/CompilationGuide

Note: you might encounter an error message using SNAP package, similar to this one:

The command could not be located because '/snap/bin' is not included in the PATH environment variable[10].

If this is the case, you can edit a file called **`/etc/environment`** and add **`/snap/bin`** in the list then restart your system.

---

10 **PATH** is an **environment variable** on Unix-like operating systems, DOS, OS/2, and Microsoft Windows, specifying a set of directories where executable programs are located

For example:

```
nano /etc/enviroment
```

Then edit the file, by adding **/snap/bin** at the end of the list:

```
PATH="/usr/local/sbin:/usr/local/bin:/usr/
sbin:/usr/bin:/sbin:/bin:/usr/games:/usr/
local/games:/snap/bin"
```

Then you can restart your system:

```
sudo reboot
```

Installing FFMPEG 4.x on Ubuntu with HomeBrew

If you want to install FFMPEG with the Homebrew Package Manager, you can do so by typing:

```
/bin/bash -c "$(curl -fsSL https://
raw.githubusercontent.com/Homebrew/install/
master/install.sh)"
```

And then by typing:

```
brew install homebrew-ffmpeg/ffmpeg/ffmpeg
```

If you need to install additional codec or protocols, you can follow the same instructions as above for MacOS X's HomeBrew.

Installing FFMPEG on Windows

As per the other platforms, on Windows machines you can install FFMPEG in several ways as described before.
Particularly with Windows 10 you can install a component named "*Windows Subsystem for Linux*".
This can give you the possibility to install a full Ubuntu release. To use the formula described in this book I reccomend you to install Ubuntu 18.04 LTS or 20.04 LTS.
For more information about installing BASH on Windows 10 please refer to this page:

https://docs.microsoft.com/en-us/windows/wsl/install-win10

Once installed the "Windows Subsystem for Linux" and a release of Ubuntu, you can then type the following line:

```
sudo apt-get install ffmpeg
```

Pre-Requisites

Although FFMPEG can be installed on a very old machine even with no graphic card installed, much faster performances can be achieved with newer machines and one or more graphic cards. If you have a configuration of one or more GPUs you can also enable a specific option called “Hardware Acceleration” on FFMPEG, and achieve an even faster experience on some operations.

However, for the sake of simplicity, this specific options and all their variants won’t be covered on this book.

If you are interested in discovering and enabling the “Hardware Acceleration” option for your FFMPEG installation please take a look at the following article:

https://trac.ffmpeg.org/wiki/HWAccelIntro

Approximately every 6 months the FFmpeg project makes a new major release. Between major releases point releases will appear that add important bug fixes but no new features[11].

[11] https://ffmpeg.org/download.html

Basic Syntax Concepts of FFMPEG

Once you have FFMPEG installed, you can run the program just by typing the word `ffmpeg` on your Terminal. The basic flow of FFMPEG is pretty straight-forward.

The program will expect an input file, might expect some options to trasform it, or to keep it as the original, and then it will create an output file.

Let's take the following very basic example.
I want to take an audio file in ".wav" format (e.g.:"mysong.wav") and convert it into an Mp3 audio file, therefore creating a new file called "mysong.mp3".

Assuming that the file "mysong.wav" is already in the current working directory, i will type a command just like the following:

```
ffmpeg -i mysong.wav mysong.mp3
```

The "**`-i`**" stands for "input". This input can be a local file in your computer or a file from a remote URL.

The file extension specified at the end of my output file, will make FFMPEG automatically choose the Mp3 format with some of the best options for that kind of file.

But having to deal with many videos or different audio files formats can be a little more complicated than this.

You might want to select a specific bitrate for that Mp3 output, or your WAV file might need to be pre-processed for example with a standard loudness normalizer filter, *before* the conversion happens.

You might end with a more complex command such as this one:

```
ffmpeg -i mysong.wav -af loudnorm -c:a mp3 -b:a 256k -ar 48000 mysong.mp3
```

This will take the same original file, but this time it will process it and then convert it to an Mp3 file.

Let's breakdown the above example:

`ffmpeg -i mysong.wav:` will process the input file "mysong.wav"

`-af loudnorm` : will apply an "Audio Filter Loudnorm" which will perform a standard audio loudness normalization

`-c:a mp3:` stands for "codec audio Mp3". This option will specify that we want to export an Mp3 file.

`-b:a 256k:` stands for "bitrate of the audio" and will produce an audio file with "Bitrate Audio at 256 Kbit/s"

`-ar 48000:` stands for "Audio Rate". This option will specify furthemore that we want to create an output file with a specific audio sampling rate (in this example 48000 KHz).

The same syntax structure is meant for video files.

Let's make an example: an Apple ProRes file ("master.mov") must be converted in h264.

The syntax will be pretty simple:

```
ffmpeg -i master.mov master.mp4
```

The extension used on the output file will trigger several default options of FFMPEG in order to produce a good quality h264 file in a mp4 container.

But we may want to have a smaller file or a particular audio option, or may need to have a stereo downmix from a 5.1 soundtrack, or again, we might want to process and output just the first 2 minutes and 30 seconds of a 3 hours long video from an URL.

Thus, we might have an example case like this:

```
ffmpeg -i https://www.dropbox.com/master.mov -t 00:02:30 -c:v h264 -b:v 3M -c:a aac -b:a 128k -ar 44100 -ac 2 master.mp4
```

-i: stands for "input"

-t: stands for "process only for a specified duration"

-c:v: stands for "codec video"

-b:v: stands for "video bitrate" which is the amount of desired video data per second, expressed in Megabit

-c:a: stands for "audio codec"

-b:a: stands for "audio bitrate", which is the amount of the desired audio data per second, expressed in kilobit

`-ar:` stands for "Audio Rate", which is also known as "sampling rate".

A *sample* is a measurement – a snapshot, if you will – at one specific time in that audio track, described in the binary language of 1s and 0s.

Repeat that measurement tens of thousands of times each second; how often that snapshot is taken represents the sample rate or sampling frequency, measured in kiloHertz, a unit meaning 1,000 times per second.

Audio CDs, for example, have a sample rate of 44.1kHz, which means that the analog signal is sampled 44,100 times per second.

`-ac 2`: stands for "2 Audio Channels (Left+Right)".

This option is useful when you have to down-mix an audio file that contains multiple tracks, such in the case of Dolby Surround 5.1/7.1 and similar audio tracks.