本文介绍 FFmpeg 的命令行使用。它可以【快速】地完成音视频的处理，包括剪辑、合并、压制、添加ass字幕；更利于批量生产地，它也可以方便地配合其他编程语言。

FFmpeg 官方网站提供免费下载。在下载之前，请打开命令行输入 ffmpeg，以确认本机是否已安装过 ffmpeg（因为它常常作为一个组件被安装，比如 ImageMagick）。

官方文档页：点击这里。

FFmpeg 命令行

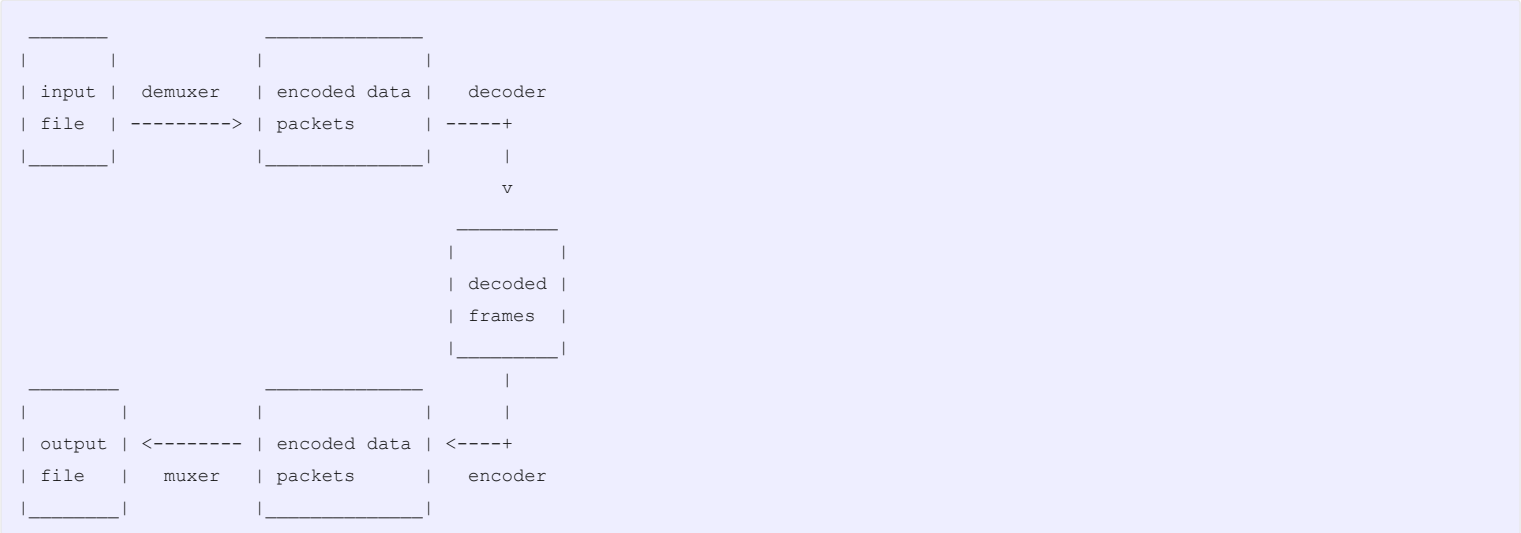
基础语法与流程

由于是命令行操作，因此语法也是很简单的。各参数以空格分隔。

```
ffmpeg [gl-opt] {[input-file-opt] -i input-url} ... {[output-file-opt] output-url} ...
```

其中，`-i` 参数表示输入参数；之后的参数是输出参数。

ffmpeg 的一般工作流，是从源文件开始，依次经过分流器、解码器、编码器、混流器，最后完成输出文件。下图是官网给出的示意：



一些基本的概念：

- 流（stream）：视频文件中，一般具有视频流与音频流，有的具有字幕流。它们需要不同的解码/编码器。想要混合/分离视频与音频，就需要混流/分流器。
- 流复制（stream）：如果某种数据流的内容不需要任何改动，那么可以直接跳过该数据流的解码与编码步骤。分离出该数据流后，直接等待参与混流即可。

ffmpeg 在有多个流的情况下，不会全部保留；默认只会选择同类流中**质量最佳的**。如果质量同样，那么选择索引号靠前的流。你可以选择手动控制流的选择，这需要额外的参数，我们下文介绍。

主要参数

下表中：

1. 以 `<>` 包裹的表示由用户具体指定；以 `[]` 包裹的表示是可选参数，可以指定也可以省略。
2. 括号内的 `i` 表示该参数用于输入流，`o` 表示用于输出流，`i/o` 表示均可，`global` 表示全局参数。
3. 关键字 `duration`，`position` 或 `offset` 满足：`[-][HH:]MM:SS[.m...]` 这种时间戳格式。或者以秒为单位的 `SS[.m...]` 格式。

常用的如下：

- `-b[:stream_specifier] (o)`：输出比特率。
- `-f <fmt> (i/o)`：指定 `fmt` 作为输入或输出的视频格式。一般会根据文件扩展名自动选择，但有时需要手动指定。
- `-i <filename> (i)`：指定 `filename` 作为源文件。
- `-y (global)`：文件存在时直接覆盖。
- `-n (global)`：文件存在时不覆盖并立即退出。
- `-bsf[:stream_specifier] <bitstream_filters> (o)`：设置比特流滤镜。`bitstream_filters` 是一个逗号分隔的滤镜列表。
- `-stream_loop <num> (i)`：指定输入流的循环次数。0 表示不循环，-1 表示无限循环。
- `-c[:stream_specifier] <codec> (i/o)`：`-c` 可写为 `-codec`。选择一个 `codec`，即编码器（输出时）或一个解码器（输入时），参与到 `stream_specifier` 指定的一个或多个流的编码/解码中。在输出时，`<codec>` 可以被指定为 `copy`，表示复制数据流。
- `-t <duration> (i/o)`：（在参数 `-i` 之前指定）工作持续 `duration` 时长。一般用于指定剪辑数据流的范围。它与 `-to` 参数相互冲突，但本参数优先。

- `-to <position> (o)`：到 `position` 位置后，终止输出。与 `-t` 参数冲突，本参数优先级低。
- `-fs <limit_size> (o)`：输出文件大小达到 `limit_size` 后停止输出，单位是 byte.
- `-ss <position> (i/o)`：（在参数 `-i` 之前指定）从 `position` 指定的位置开始工作。*注意：大多数情形下，工作起始位置是不精确的。ffmpeg会找到其前部的一个点作为真正的起始，并在结束工作后将该点与用户指定点之间的内容抛弃。然而，如果你使用了 copy 参数，这部分内容却会被保留。*
- `-sseof <position> (i/o)`：类似 `-ss` 参数，只不过是从数据流末端向前寻找 `position`。此时 0 表示数据流末。
- `-itsoffset <offset> (i)`：指定输入流以原时间戳加上 `offset` 作为其输入时间戳。
- `-metadata[:metadata_specifier] key=value (o)`：以键值对的形式设置元数据。
- `-frames[:stream_specifier] <num> (o)`：在输出 `num` 帧后停止写入。
- `-qscale[:stream_specifier] q (o)`：使用固定质量(VBR)。
- `stats (global)`：输出编码过程，是系统默认值。可以使用 `-nostats` 关闭。
- `-attach <filename> (o)`：将 `filename` 文件附加到输出文件。附件流作为文件的最后一个流，只有很少的文件类型被支持（例如字体）。

视频参数

- `-vframes <num>`：文件的总帧数。`-frames:v` 的别名。
- `-r[:steam_specifier] <fps> (i/o)`：文件的帧率。
- `-s[:steam_specifier] <size> (i/o)`：帧尺寸。参数 `size` 需要满足格式 `<width>x<height>`，例如 `320x240`。- `aspect[:steam_specifier] <asp> (o)`：宽高比，例如 `4:3`。如果使用了 `-vcodec copy`，那么指定容器的宽高比而不是视频的。
- `-vn (o)`：禁止输出视频。
- `-vcodec <codec> (o)`：设置视频编码器。`-codec:v` 的别名。
- `-pass[:stream_specifier] <n>`：选择当前编码数（1或者2），常用于二次编码的情况。在第一次编码中，音频输出往往被设置为 NULL，对于 Windows 与 Unix 系统分别是：

```
ffmpeg -i foo.mov -c:v libxvid -pass 1 -an -f rawvideo -y NUL
ffmpeg -i foo.mov -c:v libxvid -pass 1 -an -f rawvideo -y /dev/null
```

音频参数

- `-aframes <num> (o)`：文件的总帧数。`-frames:a` 的别名。
- `-ar[:stream_specifier] <freq> (i/o)`：采样率。默认输出等于输入。仅当输入文件为真实设备或者 raw 数据时，该参数才能用于输入过程。
- `-aq <q> (o)`：音频品质（VBR）。`-q:a` 的别名。
- `-ac[:stream_specifier] <channel> (i/o)`：设置音频通道数。默认输出等于输入。仅当输入文件为真实设备或者 raw 数据时，该参数才能用于输入过程。
- `-an (o)`：禁止输出音频。
- `-acode <codec> (i/o)`：设置音频的解码器或编码器。`-codec:a` 的别名。

字幕参数

- `-scodec <codec> (i/o)`：字幕解码器或编码器。`codec:s` 的别名。
- `-sn (o)`：禁止输出字幕。
- `canvas_size <size>`：设置字幕渲染区域的尺寸。

其他参数

- 以下直接在 `ffmpeg` 后使用，例如：`ffmpeg -version`。
  - `-bsfs`：可用的比特流滤镜。
  - `-h [arg]`：帮助。arg 的内容可以是：
    - `decoders`：可用的解码器。或特指：`decoder=<name>`。
    - `encoders`：可用的编码器。或特指：`encoder=<name>`。
    - `filters`：所有滤镜。或特指：`filter=<name>`。
    - `formats`：可用的分流器与混流器。或特指分流器：`demuxer=<name>`，或特指混流器：`muxer=<name>`。
  - `-protocols`：支持的协议。
  - `-version`：版本信息。

FFmpeg 实用例子

合并视频

- 参考：FFmpeg Wiki - Concatenate
- 第一种方案：将这几个视频放在一个新文件夹内，Shift 右键运行 cmd，输入（*注意：如果要保存为批处理文件，请循环变量的双写百分号。*）：

```
(for %i in (*.flv) do @echo file '%i') > mylist.txt
ffmpeg -f concat -i mylist.txt -c copy output.flv
```

- 这样速度很快也没有中间文件，原则上要求文件规格相近。
- 另一种方案：先将这几个视频无损地转为 mpegts 文件，再通过 concat 协议合并。以常见的 H.264 视频与 aac 音频为例：

```
ffmpeg -i "1.flv" -c copy -bsf:v h264_mp4toannexb -f mpegts 1.ts
ffmpeg -i "2.flv" -c copy -bsf:v h264_mp4toannexb -f mpegts 2.ts
ffmpeg -i "concat:1.ts|2.ts" -c copy -bsf:a aac_adtstoasc "All.mp4"
```

- 这种方案是早期方案，支持更广，包括非 mpeg 容器的 mpeg 编码内容（H.264, MPEG4, MPEG2, AAC, MP3等）。不过这样会产生中间文件。

分割视频

- 指定视频的起始与持续时长就可以分割视频了。下例截取了视频的前 5 秒（00:05:00），注意 `-t` 后接“截取视频段长度”而不是“截取终点时刻”：

```
ffmpeg -i "input.mp4" -ss 00:00:00 -t 5 -c copy "output.mp4"
```

- 建议使用规范的 mp4 格式文件，否则可能出现视频无法正常混流的现象。

批量格式转换

- 比如，对于数据流用 mpeg 编码的一个 flv 文件，可以这样转为 mp4 文件：

```
ffmpeg -i "input.flv" -c copy "output.mp4"
```

- 因此一个批量转换也很容易通过 for 语句实现（`%~n` 表示保留不含扩展名的文件名）：
-

```
for %i in (*.mp4) do ffmpeg -i "%i" -c copy "%~ni.flv"
```

截图

其实我觉得播放器内置的截图可能更好用。

静态图水印

下例添加 png 或其他静态格式的水印，放置在距左侧 20 像素,距顶端 40 像素的地方。水印与视频的基准点都是左上角点。

```
ffmpeg -i input.mp4 -i wm.png -filter_complex "overlay=20:40" output.mp4
```

如果要放在右下角使用 `overlay= main_w-overlay_w:main_h-overlay_h`，参数的含义应该较好理解。

如果要指定水印的大小，比如 384x216：

```
ffmpeg -i input.mp4 -i wm.png -filter_complex "[1:v]scale=384:216[wm];[0:v][wm]overlay=0:0" output.mp4
```

参数 `0:v` 表示第1个输入的视频流（本例即input.mp4的视频流），`1:v` 表示第2个输入的视频流（本例即wm.gif）。分号前的[wm]用于引用。

GIF 水印

添加 gif 水印与静态图水印有一些不同之处：

- 需要将 `ignore_loop` 参数指明为 0，表示 gif 无限循环。
- 需要用到复合过滤器 `filter_complex`。
- 需要过滤器的 `shortest=1` 选项，表示至少在一个视频流循环一次后，再终止输出。如果不加该选项，输出将无法自行停止。

一个指定 50x50 大小 GIF 水印在左上角的例子：

```
ffmpeg -y -i input.mp4 -ignore_loop 0 -i wm.gif -filter_complex "[1:v]scale=50:50[wm];[0:v][wm]overlay=0:0:shortest=1'
```

外挂字幕

将字幕作为单独的数据流（而不是混入视频流中），封装到容器内。一般对此特性有良好支持的容器是 mkv。在封装时，一般需要转为 ass 格式。

```
ffmpeg -i input.mp4 -i input.srt -c:v copy -c:a copy -c:s ass output.mkv
```

一些注意点：

- 字幕文件请用 UTF-8 编码。
- Windows 系统缺少一个字体接口，因此需要自己配置 fonts.conf 文件，放在 `%FONTCONFIG_PATH%` 这个环境用户变量里（往往需要你自己新建）。该变量应该指向 `C:\Users\用户名\`。

网上流传了一份 fonts.conf 文件内容（见附录），请复制后粘贴到你对应文件夹的 fonts.conf 文件中。

内嵌字幕

在播放器不支持独立字幕流的场合，需要将字幕混入视频流中（因此需要重编码）。

```
ffmpeg -i input.mp4 -vf subtitles=input.srt output.mp4
```

如果字幕以字幕流的形式位于一个视频文件中，可以直接调用：

```
ffmpeg -i input.mkv -vf subtitles=input.mkv output.mp4
```

同样，Windows 用户需要配置 fonts.conf 文件。

附录：Windows 的 fonts.conf

参考页面：该用户的 Github。

```
<?xml version="1.0"?>
<fontconfig>

<dir>C:\WINDOWS\Fonts</dir>

<match target="pattern">
<test qual="any" name="family"><string>mono</string></test>
<edit name="family" mode="assign"><string>monospace</string></edit>
</match>

<match target="pattern">
<test qual="all" name="family" compare="not_eq"><string>sans-serif</string></test>
<test qual="all" name="family" compare="not_eq"><string>serif</string></test>
<test qual="all" name="family" compare="not_eq"><string>monospace</string></test>
<edit name="family" mode="append_last"><string>sans-serif</string></edit>
</match>

<alias>
<family>Times</family>
<prefer><family>Times New Roman</family></prefer>
<default><family>serif</family></default>
</alias>
<alias>
<family>Helvetica</family>
<prefer><family>Arial</family></prefer>
<default><family>sans</family></default>
</alias>
<alias>
<family>Courier</family>
<prefer><family>Courier New</family></prefer>
<default><family>monospace</family></default>
</alias>
<alias>
<family>serif</family>
<prefer><family>Times New Roman</family></prefer>
</alias>
<alias>
<family>sans</family>
<prefer><family>Arial</family></prefer>
</alias>
<alias>
<family>monospace</family>
<prefer><family>Andale Mono</family></prefer>
</alias>
```

```
<match target="pattern">
<test name="family" compare="eq">
<string>Courier New</string>
</test>
<edit name="family" mode="prepend">
<string>monospace</string>
</edit>
</match>
<match target="pattern">
<test name="family" compare="eq">
<string>Courier</string>
</test>
<edit name="family" mode="prepend">
<string>monospace</string>
</edit>
</match>

</fontconfig>
```

来源： https://wklchris.github.io/FFmpeg.html#其他参数