链滴

FFmpeg Utility 帮助文档翻译

作者：HaujetZhao

原文链接：https://ld246.com/article/1595481042231

来源网站：链滴

<h2 id="描述">描述</h2>

这个文档描述了一些由 libavutil library 提供的通用特性。
<h2 id="语法">语法</h2>
FFmpeg libraries 和 tools 使用了这部分文档的语法和格式。
<h2 id="引用和转义">引用和转义</h2>
除了特殊标明的地方，FFmpeg 默认采用下述规则进行引用和转义：

' 和**** 是特殊符号（分别用于引用和转义）。除它们外，取决于包含引和转义的指定语法，可能还有其它特殊字符。
特殊字符前加上**** 则被转义成普通符号。
在分隔的字符串中，所有在**' '** 之间的符号都被当成文本。但是引用符**' '** 本身不能被引用所以你需要先把之前引用的括住，再跟一个加有转义符的**' '** 连起来。
除非被转义或者引用了，语义字符串中的空格都会被删掉。

提醒下，当使用命令行或脚本时，你可能需要二级转义，这取决于你所用的 shell 语言。
遵循上述规则时，定义在 libavutil/avstring.h 中的 av_get_token 函数可以用来分隔被引用或义的标识。
在 FFmpeg 源目录中的 tools/ffescape 工具可以被用来在脚本中自动引用或转义字符串。
<h2 id="示例">示例</h2>

将带有"'"的 Crime d'Amour 转义：

Crime d\'Amour

上面的一串文字包含一个引用符，所以 ' 需要被转义：

'Crime d'\''Amour'

前缀或后跟空格的需要引用起来：

' this string starts and ends with whitespaces '

转义和引用可以被同时使用：

' The string '\'string\'' is a string '

要包含字面的\，你需要将它引用或转义：

'c:\foo' can be written as c:\\foo
<h2 id="日期">日期</h2>
可接受的例子是：
[(YYYY-MM-DD|YYYYMMDD)[T|t|
 ]](HH:MM:SS[.m...]]])|(HHMMSS[.m...]]]))[Z]
now
如果值是"now"会取当前时间。
除非追加上了 Z（这时会使用 UTC 时间），一般时间都是当地时间。如果 year-month-day 部没有指定，默认使用当前 year-month-day。

<h2 id="持续时间">持续时间</h2>
对于持续时间，我们有两种可用语法：
[-][HH:]MM:SS[.m...]
HH 表示小时数，MM 表示分钟数，SS 表示秒数。m 表示秒后的小数点数。

或
[-]S+[.m...]
S 表示秒的数量，有一个可选的小数点后的数 m。
这两种表达式中，可选的"-"表示负的持续时间。
<h2 id="例子">例子</h2>
下列例子都表示有效的持续时间：
'55'
55 seconds
'12:03:45'
12 hours, 03 minutes and 45 seconds
'23.189'
23.189 seconds
<h2 id="视频大小">视频大小</h2>
指定源视频的大小，可以是一个 widthxheight 形式的字符串，或者表示这个大小的缩写。
下列缩写都被认可：
'ntsc'
 720x480
'pal'
 720x576
'qntsc'
 352x240
'qpal'
 352x288
'sntsc'
 640x480
'spal'
 768x576
'film'
 352x240
'ntsc-film'
 352x240
'sqcif'
 128x96
'qcif'
 176x144
'cif'
 352x288
'4cif'
 704x576
'16cif'
 1408x1152
'qqvga'
 160x120
'qvga'
 320x240
'vga'
 640x480
'svga'
 800x600
'xga'
 1024x768
'uxga'
 1600x1200
'qxga'
 2048x1536
'sxga'
 1280x1024
'qsxga'
 2560x2048
'hsxga'
 5120x4096
'wvga'
 852x480
'wxga'
 1366x768
'wsxga'
 1600x1024
'wuxga'
 1920x1200
'woxga'
 2560x1600
'wqsxga'
 3200x2048
'wquxga'
 3840x2400
'whsxga'
 6400x4096
'whuxga'
 7680x4800
'cga'
 320x200
'ega'
 640x350

'hd480'
 852x480
'hd720'
 1280x720
'hd1080'
 1920x1080
'2k'
 2048x1080
'2kflat'
 1998x1080
'2kscope'
 2048x858
'4k'
 4096x2160
'4kflat'
 3996x2160
'4kscope'
 4096x1716
'nhd'
 640x360
'hqvga'
 240x160
'wqvga'
 400x240
'fwqvga'
 432x240
'hvga'
 480x320
'qhd'
 960x540
'2kdci'
 2048x1080
'4kdci'
 4096x2160
'uhd2160'
 3840x2160
'uhd4320'
 7680x4320
<h2 id="视频速率">视频速率</h2>
指定视频的帧速率，表示为每秒生成帧的数量。应当是 frame_rate_num/frame_rate_den 的字串、一个整数、一个浮点数或是有效的帧速率简写。
下述简写是被认可的：
'ntsc
 30000/1001
'pal
 25/1
'qntsc
 30000/1001
'qpal
 25/1
'sntsc
 30000/1001
'spal
 25/1
'film
 24/1
'ntsc-film
 24000/1001
<h2 id="比例">比例</h2>
比例可以用表达式表示，或者用 numerator:denominator 的形式表示。
提醒下，负数值和无穷大（1/0）也是有效值，所以请检查好你的表达式的返回值。
未定义的值可以用"0:0"表示。
<h2 id="颜色">颜色</h2>
可以是如下所定义的颜色的名字，或是一个[0x|#]RRGGBB[AA]这样的序列，可以跟上一个@再一个字符串表示 alpha 部分。
alpha 部分可以是由"0x"跟着一个 16 进制的数组成，也可以是 0.0 到 1.0 之间的小数，后者代着透明度的数值（0 和 0x00 代表完全透明,1 和 0xff 表示不透明）。如果没有指定 alpha 值，默认为 xff。
"random"这个字符串会生成随机的颜色。
下述颜色的名字是被认可的：
'AliceBlue'
 0xF0F8FF
'AntiqueWhite'
 0xFAEBD7
'Aqua'
 0x00FFFF
'Aquamarine'
 0x7FFFD4
'Azure'
 0xF0FFFF
'Beige'
 0xF5F5DC
'Bisque'
 0xFFE4C4
'Black'
 0x000000
'BlanchedAlmond'
 0xFFEBCD
'Blue'
 0x0000FF
'BlueViolet'
 0x8A2BE2

'Brown'
 0xA52A2A
'BurlyWood'
 0xDEB887
'CadetBlue'
 0x5F9EA0
'Chartreuse'
 0x7FFF00
'Chocolate'
 0xD2691E
'Coral'
 0xFF7F50
'CornflowerBlue'
 0x6495ED
'Cornsilk'
 0xFFF8DC
'Crimson'
 0xDC143C
'Cyan'
 0x00FFFF
'DarkBlue'
 0x00008B
'DarkCyan'
 0x008B8B
'DarkGoldenRod'
 0xB8860B
'DarkGray'
 0xA9A9A9
'DarkGreen'
 0x006400
'DarkKhaki'
 0xBDB76B
'DarkMagenta'
 0x8B008B
'DarkOliveGreen'
 0x556B2F
'Darkorange'
 0xFF8C00
'DarkOrchid'
 0x9932CC
'DarkRed'
 0x8B0000
'DarkSalmon'
 0xE9967A
'DarkSeaGreen'
 0x8FBC8F
'DarkSlateBlue'
 0x483D8B
'DarkSlateGray'
 0x2F4F4F
'DarkTurquoise'
 0x00CED1
'DarkViolet'
 0x9400D3
'DeepPink'
 0xFF1493
'DeepSkyBlue'
 0x00BFFF
'DimGray'
 0x696969
'DodgerBlue'
 0x1E90FF
'FireBrick'
 0xB22222
'FloralWhite'
 0xFFFAF0
'ForestGreen'
 0x228B22
'Fuchsia'
 0xFF00FF
'Gainsboro'
 0xDCDCDC
'GhostWhite'
 0xF8F8FF
'Gold'
 0xFFD700
'GoldenRod'
 0xDAA520
'Gray'
 0x808080
'Green'
 0x008000
'GreenYellow'
 0xADFF2F
'HoneyDew'
 0xF0FFF0
'HotPink'
 0xFF69B4
'IndianRed'
 0xCD5C5C
'Indigo'
 0x4B0082
'Ivory'
 0xFFFFF0
'Khaki'
 0xF0E68C
'Lavender'
 0xE6E6FA
'LavenderBlush'
 0xFFF0F5
'LawnGreen'
 0x7CFC00
'LemonChiffon'
 0xFFFACD
'LightBlue'
 0xADD8E6
'LightCoral'
 0xF08080

'LightCyan'
 0xE0FFFF
'LightGoldenRodYellow'
 0xFAFAD2
'LightGreen'
 0x90EE90
'LightGrey'
 0xD3D3D3
'LightPink'
 0xFFB6C1
'LightSalmon'
 0xFFA07A
'LightSeaGreen'
 0x20B2AA
'LightSkyBlue'
 0x87CEFA
'LightSlateGray'
 0x778899
'LightSteelBlue'
 0xB0C4DE
'LightYellow'
 0xFFFFE0

'Lime'
 0x00FF00
'LimeGreen'
 0x32CD32
'Linen'
 0xFAF0E6
'Magenta'
 0xFF00FF
'Maroon'
 0x800000
'MediumAquaMarine'
 0x66CDAA
'MediumBlue'
 0x0000CD
'MediumOrchid'
 0xBA55D3
'MediumPurple'
 0x9370D8
'MediumSeaGreen'
 0x3CB371
'MediumSlateBlue'
 0x7B68EE
'MediumSpringGreen'
 0x00FA9A
'MediumTurquoise'
 0x48D1CC
'MediumVioletRed'
 0xC71585
'MidnightBlue'
 0x191970
'MintCream'
 0xF5FFFA
'MistyRose'
 0xFFE4E1
'Moccasin'
 0xFFE4B5
'NavajoWhite'
 0xFFDEAD
'Navy'
 0x000080
'OldLace'
 0xFDF5E6
'Olive'
 0x808000
'OliveDrab'
 0x6B8E23
'Orange'
 0xFFA500
'OrangeRed'
 0xFF4500
'Orchid'
 0xDA70D6
'PaleGoldenRod'
 0xEEE8AA
'PaleGreen'
 0x98FB98
'PaleTurquoise'
 0xAFEEEE
'PaleVioletRed'
 0xD87093
'PapayaWhip'
 0xFFEFD5
'PeachPuff'
 0xFFDAB9
'Peru'
 0xCD853F
'Pink'
 0xFFC0CB
'Plum'
 0xDDA0DD

'PowderBlue'
 0xB0E0E6
'Purple'
 0x800080
'Red'
 0xFF0000
'RosyBrown'
 0xBC8F8F
'RoyalBlue'
 0x4169E1
'SaddleBrown'
 0x8B4513
'Salmon'
 0xFA8072
'SandyBrown'
 0xF4A460
'SeaGreen'
 0x2E8B57
'SeaShell'
 0xFFF5EE
'Sienna'
 0xA0522D
'Silver'
 0xC0C0C0
'SkyBlue'
 0x87CEEB
'SlateBlue'
 0x6A5ACD
'SlateGray'
 0x708090
'Snow'
 0xFFFAFA
'SpringGreen'
 0x00FF7F
'SteelBlue'
 0x4682B4
'Tan'
 0xD2B48C
'Teal'
 0x008080
'Thistle'
 0xD8BFD8
'Tomato'
 0xFF6347
'Turquoise'
 0x40E0D0
'Violet'
 0xEE82EE
'Wheat'
 0xF5DEB3
'White'
 0xFFFFFF
'WhiteSmoke'
 0xF5F5F5
'Yellow'
 0xFFFF00
'YellowGreen'
 0x9ACD32
<h2 id="声道布局">声道布局</h2>
声道布局表示在多声道音频流中声道的物理空间布局。要表示一个声道布局，FFmpeg 使用一种
殊的语法。
单独的声道被由下表给出的 id 区分：
'FL'
 front left
'FR'
 front right
'FC'
 front center
'LFE'
 low frequency
'BL'
 back left
'BR'
 back right
'FLC'
 front left-of-center
'FRC'
 front right-of-center
'BC'
 back center
'SL'
 side left
'SR'
 side right
'TC'
 top center
'TFL'
 top front left
'TFC'
 top front center
'TFR'
 top front right
'TBL'
 top back left
'TBC'
 top back center
'TBR'
 top back right
'DL'
 downmix left
'DR'
 downmix right
'WL'
 wide left

'WR'
 wide right
'SDL'
 surround direct left
'SDR'
 surround direct right
'LFE2'
 low frequency 2
标准声道布局组合可以由如下的标识符表示：
'mono'
 FC
'stereo'
 FL+FR
'2.1'
 FL+FR+LFE
'3.0'
 FL+FR+FC
'3.0(back)'
 FL+FR+BC
'4.0'
 FL+FR+FC+BC
'quad'
 FL+FR+BL+BR
'quad(side)'
 FL+FR+SL+SR
'3.1'
 FL+FR+FC+LFE
'5.0'
 FL+FR+FC+BL+BR
'5.0(side)'
 FL+FR+FC+SL+SR
'4.1'
 FL+FR+FC+LFE+BC
'5.1'
 FL+FR+FC+LFE+BL+BR
'5.1(side)'
 FL+FR+FC+LFE+SL+SR
'6.0'
 FL+FR+FC+BC+SL+SR
'6.0(front)'
 FL+FR+FLC+FRC+SL+SR
'hexagonal'
 FL+FR+FC+BL+BR+BC
'6.1'
 FL+FR+FC+LFE+BC+SL+SR
'6.1'
 FL+FR+FC+LFE+BL+BR+BC
'6.1(front)'
 FL+FR+LFE+FLC+FRC+SL+SR
'7.0'
 FL+FR+FC+BL+BR+SL+SR
'7.0(front)'
 FL+FR+FC+FLC+FRC+SL+SR
'7.1'
 FL+FR+FC+LFE+BL+BR+SL+SR
'7.1(wide)'
 FL+FR+FC+LFE+BL+BR+FLC+FRC
'7.1(wide-side)'
 FL+FR+FC+LFE+FLC+FRC+SL+SR
'octagonal'
 FL+FR+FC+BL+BR+BC+SL+SR
'hexadecagonal'
 FL+FR+FC+BL+BR+BC+SL+SR+WL+WR+TBL+TBR+TBC+TFC+TFL
TFR
'downmix'
 DL+DR
一个自定义音频布局可以用一系列术语表示，由"+"或"|"分隔。每个术语可以是：

标准声道布局的名字（例如'mono', 'stereo', '4.0', 'quad', '5.0'）。
单独一个声道的名字（例如'FL', 'FR', 'FC', 'LFE'）。
十进制数表示的多个声道，后接一个'c'，产生这个声道数表示的默认声道布局（详见 av_get_def
ult_channel_layout 函数）。提醒下，并不是所有声道数量都有默认音频布局的。
一个音频布局的马甲名，在 libavutil/channel_layout.h 中的由'0x'开头的 16 进制（详见 AV_C
_*）宏。

在 libavutil
 53 版本之前，在一个指定数字后跟一个'c'是可选的，但现在它是被要求的，
在通过一个十进制数也可以表示一个声道布局马甲名（如果而且只有如果后面没有跟'c'或'C'时）。</
>
另外请参阅在 libavutil/channel_layout.h 中定义的 av_get_channel_layout 函数。
<h2 id="表达式值的计算">表达式值的计算</h2>
当计算一个算数表达式时，FFmpeg 会使用通过 libavutil/channel_layout.h 界面实现的内置公
计算器。
一个表达式可包含一元运算符、二元运算符、常数、函数。
两个表达式 expr1 和 expr2 可以被包含在另一个表达式之内"expr1;expr2"。expr1 和 expr2 的
被依次计算，并且新表达式会计算到 expr2 的值。
下列是可用的二元运算符：+，-，*，/

下列是可用的一元运算符：+，-
下列是可用的函数：
abs(x)
 计算 x 的绝对值
acos(x)
 计算 x 的反余弦值。
asin(x)
 计算 x 的反余弦值。
atan(x)
 计算 x 的反正切值。
atan2(x, y)
 计算 x/y 的反余切值。
between(x, min, max)
 若 x 出于 min 和 max 的闭区间，返回 1，否则返回 0.
bitand(x, y)
 bitor(x, y)
 将 x 和 y 进行按位与或运算。运算前，x 和 y 会先被转换成数。提醒下，转换成整数和转换成浮点数都会损失精度。注意下非期望的结果的特大数（通常大于 2^3）。
ceil(expr)
 将 expr 的值向前推进，直到碰到离它最近的整数。例如 ceil(1.5)是 2。
clip(x, min, max)
 返回在 min 和 max 之间的 x 数组中的值（低于 min 和高于 max 的值剪裁掉）。
cos(x)
 计算 x 的正弦值
cosh(x)
 计算 x 的双曲余弦。
eq(x, y)
 如果 x 和 y 等价，则返回 1，否则返回 0。
exp(x)
 计算 x 的指数（基 e，欧拉数）。
floor(expr)
 将表达式 exr 的值向下舍入到最近的整数。例如，floor(2.5)=2。
gauss(x)
 计算 x 的高斯函数，对应于 exp(-x*x/2) / sqrt(2*PI)。
gcd(x, y)
 返回 x 和 y 的最大公约数..如果 x 和 y 都是 0，或者两者都小于零，则行为是未义的。
gt(x, y)
 如果 x 大于 y，则返回 1，否则返回 0。
gte(x, y)
 如果 x 大于或等于 y，则返回 1，否则返回 0。hypot(x, y)
 此函数类似于同的 C 函数；它返回"sqrt（x*x+y*y）"、长度为 x 和 y 的直角三角形的斜边的长度或点 t(x,y）离原点距离。
if(x, y)
 求 x 的值，若结果非零，则返回 y 值的计算结果，否则返回 0。
if(x, y, z)
 求 x 值，若结果为非零则返回 y 的值，否则返回 z 的值。
<script async src="https://pagead2.googlesyndication.com/pagead/js/adsbygoogle.js?client=a-pub-5357405790190342" crossorigin="anonymous"></script> <!-- 黑客派PC帖子内嵌 -->
<ins class="adsbygoogle" style="display:block" data-ad-client="ca-pub-5357405790190342" data-ad-slot="8316640078" data-ad-format="auto" data-full-width-responsive="true"></in>
<script>
    (adsbygoogle = window.adsbygoogle || []).push({});
</script>
ifnot(x, y)
 求 x 值，若结果为零则返回 y 的求值结果，否则返回 0。
ifnot(x, y, z)
 求 x 值，若结果为零则返回 y 的值，否则返回 z 的评值。
isinf(x)
 如果 x 是正负无穷大，就返回 1，否则返回 0。
isnan(x)
 如果 x 是 NAN 就返回 1，否则返回 0。
ld(var)
 将之前通过 st(var, expr)
 设置的内置函数 var 的值载入。这个函数会返回被入的值。ld=load
lerp(x, y, z)
 返回 z 个 x 和 y 之间的线性差值。（linear interpolation）
log(x)
 计算 x 的 log 值
lt(x, y)
 如果 x 小于 y，则返回 1，否则返回 0。
lte(x, y)
 如果 x 小于或等于 y，则返回 1，否则返回 0。
max(x, y)
 返回 x 到 y 之间的最大值。
min(x, y)
 返回 x 到 y 之间的最小值。
mod(x, y)
 计算 x 除以 y 的余数。
not(expr)
 如果 expr 为零，则返回 1.0，否则返回 0.0。
pow(x, y)
 计算 x 的 y 次方。
print(t)
 print(t, l)
 用 loglevel
 'l'打印't'的值。如果没有指定'l'，则使用默认日级别。返回打印的值。
random(x)
 返回 0.0 到 1.0 之间的伪随机值。x 是内部变量的索引，将用于保存 state/see

。
root(expr, max)
 在 0 到 max 这个区间内，找到一个输入值，使得由 expr 参数 ld(0)表示函数为 0。
表达式中的表达式必须表示连续函数，否则结果不会被定义。
ld(0)被用于表示函数输入值，也就是给定表达式会被用各种能让表达式通过 ld(0)的输入值计算次。当计算值的结果是 0 时，就会返回对应的输入值。
round(expr)
 将表达式 exr 的值舍入最近的整数。例如，"round(1.5)"是"2"。
sgn(x)
 计算 x 的 sign 值。
sin(x)
 计算 x 的 sin 值。
sinh(x)
 计算 x 的双曲 sin 值
sqrt(expr)
 计算 expr 的平方根。
squish(x)
 计算表达式 1 / (1 + exp(4 * x)) 的值。
st(var, expr)
将 expr 表达式的值储存在一个内部变量中，var 指定了存放值的变量的数量，取值 0 到 9.这个数会返回存储在内置变量的值。目前变量不在在表达式之间共享。
tan(x)
 计算 x 的正切值。
tanh(x)
 计算 x 的双曲正切值。
taylor(expr, x)
 taylor(expr, x, id)
 给定一个表示函数在 0 出第 ld(id)阶导数的表达式原函数在 x 处求泰勒级数。
当级数不收敛时，结果是未定义的。
使用 ld(id)表示 expr 中的导数阶数，这意味着给定的表达式将被用各种能使表达式通过 ld(id)的入值计算多次。如果没有指定 id，则假定其为 0。
提醒下，当你有在 y 处的导数，而不是 0 时，就可以用 taylor(expr, x-y)。
time(0)
 以秒为单位返回当前（挂钟的）时间，也就是今天过了多少秒了。
trunc(expr)
 返回表达式 exr 的值附近离零最近的整数。例如，"trunc(-1.5)"是"-1.0"。</p

while(cond, expr)
 当表达式 cond 为非零时，计算表达式 expr，并返回最后一个 expr 计的值，若 cond 为假，则返回 NAN。
下列是可用的常数值：
PI
 约等于 3.14
E
 欧拉数 exp(1)，约等于 2.718
PHI
 黄金比例 (1 + sqrt(5)) / 2，约等于 1.618
假设一个表达式为真，并且有一个非零值，注意下：
* 用起来和 AND 一样
+ 用起来和 OR 一样
例如：
if (A AND B) then C
其实与这个相同：
if(A*B, C)
在你的 C 代码里，你可以扩充一元和二元方程的列表，然后定义被认可的常数，让它们被你的达式认可。
计算过程也认可国际系统单位前缀。如果在前缀后加上了'i'，就会使用二元前缀，意思是以 1024 为底的指数，而不是以 1000 为底的指数。后缀'B'会使得实际数值为 8 倍，可以被单独使用或跟在单前缀后面。例如：KB、MiB、G、B。
下列是可用的国际单位前缀，代表着 10 或 2 的指定次方。
y
 10^-24 / 2^-80
z
 10^-21 / 2^-70
a
 10^-18 / 2^-60
f
 10^-15 / 2^-50
p
 10^-12 / 2^-40
n
 10^-9 / 2^-30
u
 10^-6 / 2^-20
m
 10^-3 / 2^-10
c
 10^-2

d
 10^-1
h
 10^2
k
 10^3 / 2^10
K
 10^3 / 2^10
M
 10^6 / 2^20
G
 10^9 / 2^30
T
 10^12 / 2^40
P
 10^15 / 2^40
E
 10^18 / 2^50
Z
 10^21 / 2^60
Y
 10^24 / 2^70
<h2 id="另外请参阅">另外请参阅</h2>
ffmpeg, ffplay, ffprobe, libavutil
<h2 id="作者们">作者们</h2>
原英文文档作者为 FFmpeg 的开发者们。本文由淳帅二代翻译。
若要查看关于作者权的详情信息，请通过在 FFmpeg 的源文件夹输入"git log"命令，或进入 http://source.FFmpeg.org 访问在线库，来参阅这个项目（git://source.FFmpeg.org/FFmpeg）的 Git
 历史。
源代码树的"MAINTAINERS"文件中列出了所有特定组件的维护者。
该文档的英文原版生成于 2020 年 2 月 16 号。翻译完成于 2020 年 2 月 16 号。