

```

*****
* QuickTime Movie Format *
*****

- values use big endian (network) byte order
- general terms: integer = signed value
- general values: byte/char/octet = 8-bit value; short/word = 16-bit value; long = 32-bit value
- fixed point values: value made up of an integer for whole numbers and an unsigned value for the decimal
- binary values: base-2 long unsigned values (values from 0 and 1)
- octal values: base-8 long unsigned values (values from 0 through to 7)
- decimal values: base-10 long unsigned values (values from 0 through to 9)
- hexadecimal (hex) values: base-16 long unsigned values (values from 0 to 9 and A to F)
- atom offsets: values relative to atoms only and are used to skip to the next atom
- sample chunk offsets: values relative to the file's data fork
- long mac file version: byte hex version + byte hex revision + byte hex revision stage + byte hex non-final
number
- mac version revision stages: development = 0x20 ; alpha = 0x40 ; beta = 0x60 ; release = 0x80

FILE INFO

Suffixes = ".mov", ".qt"; Mac OS Type = "MooV"; Mac OS Creator = "TVOD"; MIME="video/quicktime"

Standard single fork binary file that only uses a resource fork on HFS/HFS+ volumes to store mac
specific file info, movie previews and can store the file's header.

Unknown atoms can be safely skipped over, most atoms can be in any order and all lowercase
long ASCII text strings used for atom names/types are reserved for use by Apple.

FILE MOVIE DATA

Note: if any atom grows in excess of 2^32 bytes (> 4.2 GB), the atom size can be extended in increments
of 64 bits (18.4 EB). By setting the atom size to 1 and appending a new 64 bit atom size. This is why
empty 'wide' atoms may be found on either side of this atom header for future expansion in QT 3 and later.

* 8+ bytes movie data atom = long unsigned offset + long ASCII text string 'mdat'
-> 8 bytes larger file offset place holder atom = long unsigned offset set to 8 + long ASCII text string
'wide'
OR
-> 8 bytes wider mdat atom offset = 64-bit unsigned offset if mdat standard offset set to 1

-> Sample data = hex dump. Movies with multiple tracks have sample data interleaved unless preloaded.
- if atom begins with video/audio track sample data (ie. not MPEG or Flash) then prefix with:
-> 8 bytes movie data start point = long value set to zero + long ASCII text string 'mdat'
- MPEG samples are stored as complete video, audio, or system streams unaltered
- JPEG YUV scans are stored as standard JFIFs
- Other sample data structures are documented in Apple's QT format dev docs.

UNUSED SPACE OR DATA TO BE DELETED/REUSED WITHIN FILE

* 8+ bytes free space (current) atom = long unsigned offset + long ASCII text string 'free'

* 8+ bytes skip over (older) atom = long unsigned offset + long ASCII text string 'skip'

* 8+ bytes widen (lengthen) file atom = long unsigned offset + long ASCII text string 'wide'

OPTIONAL/UNOFFICIAL FILE PREVIEW DATA

Note: the movie preview was and still is stored in the resource fork as type 'pnot'; ID any
on the Macintosh platform. But could just as easily be stored in a single file fork.
The advantage of using another file fork is that the preview image data can be updated more easily.

* 8+ bytes optional preview location, not movie data atom = long unsigned offset + long ASCII text string 'pnot'
-> 4 bytes modified mac date = long unsigned value in seconds since beginning 1904 to 2040
-> 2 bytes version = short integer value (none = 0)
-> 4 bytes resource/atom type = long ASCII text string
- types are image preview = 'PICT' ; use movie header = 'moov'
-> 2 bytes resource/atom id = short integer value
- IDs are image preview = id value ; use movie header = -1

* 8+ bytes optional preview data for single fork atom = long unsigned offset + long ASCII text string 'PICT'
-> image data = quickdraw picture resource format hex dump

FILE MOVIE HEADER

Note: the header is safer when stored at the beginning of the file or in the
resource fork as type 'moov'; ID any. The advantage of using another file fork is
that the header can be lengthened without recalculating the sample offsets or new header
must be written at the end of the file.

* 8+ bytes movie resource atom = long unsigned offset + long ASCII text string 'moov'

* 8+ bytes optional movie data reference atom = long unsigned offset + long ASCII text string 'mdra'
- if this is used no other atoms should be present

* 8+ bytes data reference atom = long unsigned offset + long ASCII text string 'dref'
-> 4 bytes reference type name = long ASCII text string

```

```

- types are file alias = 'alis'; resource alias = 'rsrc'; url c string = 'url '
-> 4 bytes reference version/flags = byte hex version (current = 0) + 24-bit hex flags
- some flags are external data = 0x000000; internal data = 0x000001
-> mac os file alias record structure
OR
-> mac os file alias record structure plus resource info
OR
-> url c string = ASCII text string
-> 1 byte url c string end = byte value set to zero

* 8+ bytes compressed moov atom = long unsigned offset + long ASCII text string 'cmov'
- if this is used no other atoms should be present as this is for an entire compressed movie resource

* 8+ bytes data compression atom = long unsigned offset + long ASCII text string 'dcom'
-> 4 bytes compression code = long ASCII text string
- compression codes are Deflate = 'zlib' ; Apple Compression = 'adec'

* 8+ bytes compressed moov data atom = long unsigned offset + long ASCII text string 'cmvd'
-> 4 bytes uncompressed size = long unsigned value
-> entire compressed movie 'moov' resource = hex dump

* 8+ bytes reference movie record atom = long unsigned offset + long ASCII text string 'rmra'
- if this atom is used it must come first within the movie resource atom

* 8+ bytes reference movie descriptor atom = long unsigned offset + long ASCII text string 'rmda'

* 8+ bytes reference movie data reference atom = long unsigned offset + long ASCII text string 'rdrf'
-> 4 bytes reference version/flags = byte hex version (current = 0) + 24-bit hex flags
- some flags are external data = 0x000000; internal data = 0x000001
-> 4 bytes reference type name = long ASCII text string (if internal = 0)
- types are file alias = 'alis'; resource alias = 'rsrc'; url c string = 'url '
-> 4+ bytes reference data = long unsigned length
-> mac os file alias record structure
OR
-> mac os file alias record structure plus resource info
OR
-> url c string = ASCII text string
-> 1 byte url c string end = byte value set to zero

* 8+ bytes optional reference movie quality atom = long unsigned offset + long ASCII text string 'rmqu'
-> 4 bytes queue position = long unsigned value from 100 to 0

* 8+ bytes optional reference movie cpu rating atom = long unsigned offset + long ASCII text string
'rmcs'
-> 4 bytes reserved flag = byte hex version + 24-bit hex flags (current = 0)
-> 2 bytes speed rating = short unsigned value from 500 to 100

* 8+ bytes optional reference movie version check atom = long unsigned offset + long ASCII text string
'rmvc'
-> 4 bytes flags = byte hex version + 24-bit hex flags (current = 0)
-> 4 bytes gestalt selector = long ASCII text string (eg. quicktime = 'qtim')

-> 4 bytes gestalt min value = long hex value (eg. QT 3.02 mac file version = 0x03028000)
-> 4 bytes gestalt no value = long value set to zero
OR
-> 4 bytes gestalt value mask = long hex mask
-> 4 bytes gestalt value = long hex value

-> 2 bytes gestalt check type = short unsigned value (min value = 0 or mask = 1)

* 8+ bytes optional reference movie component check atom = long unsigned offset + long ASCII text
string 'rmcd'
-> 4 bytes flags = byte hex version + 24-bit hex flags (current = 0)
-> 8 bytes component type/subtype = long ASCII text string + long ASCII text string
- (eg. Timecode Media Handler = 'mhlrtmcd')
-> 4 bytes component manufacturer = long ASCII text string (eg. Apple = 'appl' or 0)
-> 4 bytes component flags = long hex flags (none = 0)
-> 4 bytes component flags mask = long hex mask (none = 0)
-> 4 bytes component min version = long hex value (none = 0)

* 8+ bytes optional reference movie data rate atom = long unsigned offset + long ASCII text string
'rmdr'
-> 4 bytes flags = byte hex version + 24-bit hex flags (current = 0)
-> 4 bytes data rate = long integer bit rate value
- common analog modem rates are 1400; 2800; 3300; 5600
- common broadband rates are 5600; 11200; 25600; 38400; 51200; 76800; 100000
- common high end broadband rates are T1 = 150000; no limit/LAN = 0x7FFFFFFF

* 8+ bytes optional reference movie language atom = long unsigned offset + long ASCII text string
'rmla'
-> 4 bytes flags = byte hex version + 24-bit hex flags (current = 0)
-> 2 bytes mac language = short unsigned language value (english = 0)

* 8+ bytes optional reference movie alternate group atom = long unsigned offset + long ASCII text
string 'rmag'
-> (structure was not provided in MoviesFormat.h of the 4.1.2 win32 sdk)
-> 4 bytes flags = long value set to zero

```

```

-> 2 bytes alternate/other = short integer track id value (none = 0)

* 8+ bytes movie header atom = long unsigned offset + long ASCII text string 'mvhd'
-> 4 bytes version/flags = byte hex version + 24-bit hex flags (current = 0)
-> 4 bytes created mac date = long unsigned value in seconds since beginning 1904 to 2040
-> 4 bytes modified mac date = long unsigned value in seconds since beginning 1904 to 2040
-> 4 bytes time scale = long unsigned time unit (default = 600)
-> 4 bytes duration = long unsigned time length (in time units)
-> 4 bytes decimal user playback speed = long fixed point rate (normal = 1.0)
-> 2 bytes user volume = short unsigned level (mute = 0; 100% = 255; 300% max = 767)
-> 10 bytes reserved = 5 * short values set to zero
-> 4 bytes decimal window geometry matrix value A = long fixed point width scale (normal = 1.0)
-> 4 bytes decimal window geometry matrix value B = long fixed point width rotate (normal = 0.0)
-> 4 bytes decimal window geometry matrix value U = long fixed point width angle (normal = 0.0)
-> 4 bytes decimal window geometry matrix value C = long fixed point height rotate (normal = 0.0)
-> 4 bytes decimal window geometry matrix value D = long fixed point height scale (normal = 1.0)
-> 4 bytes decimal window geometry matrix value V = long fixed point height angle (normal = 0.0)
-> 4 bytes decimal window geometry matrix value X = long fixed point position (left = 0.0)
-> 4 bytes decimal window geometry matrix value Y = long fixed point position (top = 0.0)
-> 4 bytes decimal window geometry matrix value W = long fixed point divider scale (normal = 1.0)
-> 8 bytes preview = long unsigned start time + long unsigned time length (in time units)
-> 4 bytes still poster = long unsigned frame time (in time units)
-> 8 bytes selection time = long unsigned start time + long unsigned time length (in time units)
-> 4 bytes current time = long unsigned frame time (in time units)
-> 4 bytes next/new track id = long integer value (single track = 2)

* 8+ bytes optional clipping (mask) atom = long unsigned offset + long ASCII text string 'clip'

* 8+ bytes clipping region atom = long unsigned offset + long ASCII text string 'crgn'
-> 2 bytes region size = short unsigned box size
-> 8 bytes region boundary = long fixed point x value + long fixed point y value
-> QuickDraw Region Data = hex dump

* 8+ bytes track atom = long unsigned offset + long ASCII text string 'trak'

* 8+ bytes track header atom = long unsigned offset + long ASCII text string 'tkhd'
-> 1 bytes version = byte hex version (current = 0)
-> 3 bytes flags = 24-bit unsigned flags
    - sum of TrackEnabled = 1; TrackInMovie = 2; TrackInPreview = 4; TrackInPoster = 8
-> 4 bytes created mac date = long unsigned value in seconds since beginning 1904 to 2040
-> 4 bytes modified mac date = long unsigned value in seconds since beginning 1904 to 2040
-> 4 bytes track id = long integer value (first track = 1)
-> 8 bytes reserved = 2 * long value set to zero
-> 4 bytes duration = long unsigned time length (in time units)
-> 8 bytes reserved = 2 * long value set to zero
-> 2 bytes video layer = short integer position (middle = 0; negatives are in front)
-> 2 bytes alternate/other = short integer track id (none = 0)
-> 2 bytes track audio volume = short unsigned level (mute = 1; 100% = 256; 200% max = 512)
-> 2 bytes reserved = short value set to zero
-> 4 bytes decimal video geometry matrix value A = long fixed point width scale (normal = 1.0)
-> 4 bytes decimal video geometry matrix value B = long fixed point width rotate (normal = 0.0)
-> 4 bytes decimal video geometry matrix value U = long fixed point width angle (normal = 0.0)
-> 4 bytes decimal video geometry matrix value C = long fixed point height rotate (normal = 0.0)
-> 4 bytes decimal video geometry matrix value D = long fixed point height scale (normal = 1.0)
-> 4 bytes decimal video geometry matrix value V = long fixed point height angle (normal = 0.0)
-> 4 bytes decimal video geometry matrix value X = long fixed point position (left = 0.0)
-> 4 bytes decimal video geometry matrix value Y = long fixed point position (top = 0.0)
-> 4 bytes decimal video geometry matrix value W = long fixed point divider scale (normal = 1.0)
-> 8 bytes decimal video frame size = long fixed point width + long fixed point height

* 8+ bytes optional clipping (mask) atom = long unsigned offset + long ASCII text string 'clip'
- see moov clipping atom above

* 8+ bytes optional matte (video overlay) atom = long unsigned offset + long ASCII text string 'matt'

* 8+ bytes compressed matte atom = long unsigned offset + long ASCII text string 'kmat'
-> 4 bytes version/flags = byte hex version + 24-bit hex flags (current = 0)
-> Matte Image Description Structure (similar to Media Sample Description Table)
-> Matte Data = hex dump

* 8+ bytes optional edits (# of external tracks) atom = long unsigned offset + long ASCII text string
'edts'
- if tracks are of different time lengths this atom is needed to maintain media sync.

* 8+ bytes optional edit list atom = long unsigned offset + long ASCII text string 'elst'
-> 4 bytes version/flags = byte hex version + 24-bit hex flags (current = 0)
-> 4 bytes number of edits = long unsigned total (default = 1)
-> 8 bytes edit time = long unsigned time length + long unsigned start time (in time units)
-> 4 bytes decimal playback speed = long fixed point rate (normal = 1.0)

* 8+ bytes optional preload atom = long unsigned offset + long ASCII text string 'load'
-> 8 bytes preload time = long unsigned start time + long unsigned time length (in time units)
-> 4 bytes flags = long integer value
    - flags are PreloadAlways = 1 or TrackEnabledPreload = 2
-> 4 bytes default hints flags = long hex data play options
    - flags are KeepInBuffer = 0x00000004; HighQuality = 0x00000100; SingleFieldVideo = 0x00100000

```

```

* 8+ bytes optional track references atom = long unsigned offset + long ASCII text string 'tref'

* 8+ bytes type of reference atoms = long unsigned offset + long ASCII text string
-> atom types are timecode = 'tmcd'; chapterlist = 'chap'; stream hint = 'hint'
-> atom types are transcript (text) = 'scpt'; synchronization (video/audio) = 'sync'
-> atom types are non-primary source (used in other track) = 'ssrc'
-> 4+ bytes Track IDs = long integer track numbers (Disabled Track ID = 0)

* 8+ bytes optional non-primary source input map atom = long unsigned offset + long ASCII text string
'imap'

* 8+ bytes input atom = long unsigned offset + long ASCII text string 0x0000 + 'in'
-> 4 bytes atom ID = long integer atom reference (first ID = 1)
-> 2 bytes reserved = short value set to zero
-> 2 bytes number of internal atoms = short unsigned count
-> 4 bytes reserved = long value set to zero

* 8+ bytes input type atom = 32-bit integer unsigned + long ASCII text string 0x0000 + 'ty'
-> 4 bytes type modifier name = long integer value
-> name values are matrix = 1; clip = 2; volume = 3; audio balance = 4
-> name values are graphics mode = 5; matrix object = 6
-> name values are graphics mode object = 7; image type = 'vide'

* 8+ bytes object ID atom = long unsigned offset + long ASCII text string 'obid'
-> 4 bytes object ID = long integer value

* 8+ bytes media atom = long unsigned offset + long ASCII text string 'mdia'

* 8+ bytes media header atom = long unsigned offset + long ASCII text string 'mdhd'
-> 4 bytes version/flags = byte hex version + 24-bit hex flags (current = 0)
-> 4 bytes created mac date = long unsigned value in seconds since beginning 1904 to 2040
-> 4 bytes modified mac date = long unsigned value in seconds since beginning 1904 to 2040
-> 4 bytes time scale = long unsigned media time unit (audio = sample per sec. rate)
-> 4 bytes duration = long unsigned media time length (in media time units)
-> 2 bytes mac language = short unsigned content language value (english = 0)
-> 2 bytes quality = short integer playback quality value (normal = 0)

* 8+ bytes handler reference atom = long unsigned offset + long ASCII text string 'hdlr'
- this atom must be toward the start of the media atom
-> 4 bytes version/flags = byte hex version + 24-bit hex flags (current = 0)
-> 8 bytes type/subtype = long ASCII text string + long ASCII text string
  - (eg. Video Media Handler = 'mhlrvide' ; Audio Media Handler = 'mhlrsoun')
-> 4 bytes manufacturer reserved = long ASCII text string (eg. Apple = 'appl' or 0)
-> 4 bytes component reserved flags = long hex flags (none = 0)
-> 4 bytes component reserved flags mask = long hex mask (none = 0)
-> 1 byte component name string length = byte unsigned length (no name = zero length string)
-> component type name ASCII string (eg. "Media Handler")
  - note: the MPEG-4 spec uses a C string instead of the above Pascal string

* 8+ bytes media information atom = long unsigned offset + long ASCII text string 'minf'

* 8+ bytes video media info header atom = long unsigned offset + long ASCII text string 'vmhd'
-> 4 bytes version/flags = byte hex version + 24-bit hex flags
  - version = 0 ; flags = 0x000001 for QT 1.0 or zero for newer QT
-> 2 bytes QuickDraw graphic mode = short hex type
  - mode types are copy = 0x0000; dither copy = 0x0040; straight alpha = 0x0100
  - mode premultiplied types are white alpha = 0x101; black alpha = 0x102
  - mode types are composition dither copy = 0x0103
  - mode color types are blend = 0x0020; transparent = 0x0024; straight alpha blend = 0x0104

-> 2 bytes graphic mode no color = short value set to zero
OR
-> 6 bytes graphic mode color = 3 * short unsigned QuickDraw RGB color values
OR
* 8+ bytes sound media info header atom = long unsigned offset + long ASCII text string 'smhd' or
-> 4 bytes version/flags = byte hex version + 24-bit hex flags (current = 0)
-> 2 bytes audio balance = short fixed point value
  - balance scale is left = negatives ; normal = 0 ; right = positives
-> 2 bytes reserved = short value set to zero
OR
* 8+ bytes generic media info header atom = long unsigned offset + long ASCII text string 'gmhd'
- used for MPEG, Timecode, MIDI, Flash, or any media that doesn't fit into the above

* 8+ bytes generic media info atom = long unsigned offset + long ASCII text string 'gmin'
-> 4 bytes version/flags = byte hex version + 24-bit hex flags (current = 0)
-> 2 bytes QuickDraw graphic mode = short hex type
  - mode types are copy = 0x0000; dither copy = 0x0040; straight alpha = 0x0100
  - mode premultiplied types are white alpha = 0x101; black alpha = 0x102
  - mode types are composition dither copy = 0x0103
  - mode color types are blend = 0x0020; transparent = 0x0024; straight alpha blend = 0x0104

-> 2 bytes graphic mode no color = short value set to zero
OR
-> 6 bytes graphic mode color = 3 * short unsigned RGB color values

-> 2 bytes audio balance = short fixed point value
  - balance scale is left = negatives ; normal = 0 ; right = positives

```

```

-> 2 bytes reserved = short value set to zero

* 8+ bytes optional handler reference atom = long unsigned offset + long ASCII text string 'hdlr'
-> 4 bytes version/flags = byte hex version + 24-bit hex flags (current = 0)
-> 8 bytes type/subtype = long ASCII text string + long ASCII text string
  - (eg. Alias Data Handler = 'dhlralis' ; URL Data Handler = 'dhlrurl ')
-> 4 bytes manufacturer reserved = long ASCII text string (eg. Apple = 'appl' or 0)
-> 4 bytes component reserved flags = long hex flags (none = 0)
-> 4 bytes component reserved flags mask = long hex mask (none = 0)
-> 1 byte component name string length = byte unsigned length (no name = zero length string)
-> component type name ASCII string (eg. "Data Handler")

* 8+ bytes data information atom = long unsigned offset + long ASCII text string 'dinf'

  * 8+ bytes data reference atom = long unsigned offset + long ASCII text string 'dref'
  -> 4 bytes version/flags = byte hex version + 24-bit hex flags (current = 0)
  -> 4 bytes number of references = long unsigned total (minimum = 1)

    * 8+ bytes reference type atom = long unsigned offset + long ASCII text string
    - atom types are file alias = 'alis'; resource alias = 'rsrc'; url c string = 'url '
    -> 4 bytes version/flags = byte hex version (current = 0) + 24-bit hex flags
    - some flags are external data = 0x000000; internal data = 0x000001
    -> mac os file alias record structure points to external data
    OR
    -> mac os file alias record structure plus resource info points to external data
    OR
    -> url c string = ASCII text string points to external data
    -> 1 byte url c string end = byte value set to zero

  * 8+ bytes sample table atom = long unsigned offset + long ASCII text string 'stbl'

    * 8+ bytes sample description atom = long unsigned offset + long ASCII text string 'stds'
    -> 4 bytes version/flags = byte hex version + 24-bit hex flags (current = 0)
    -> 4 bytes number of descriptions = long unsigned total (default = 1)

      -> 4 bytes description length = long unsigned length (86 byte minimum for video)
      -> 4 bytes description data format = long ASCII text string
      -> 6 bytes reserved = 48-bit value set to zero
      -> 2 bytes data reference index = short unsigned index from 'dref' atom
      - there are other sample descriptions available in the Apple QT format dev docs

      -> 2 bytes video/audio encoding version = short hex version
      - default = 0; audio data size before decompression = 1
      -> 2 bytes video/audio encoding revision level = byte hex version
      - default = 0; video can revise this value
      -> 4 bytes video/audio encoding vendor = long ASCII text string
      - video uses Apple = 'appl'; audio = 0

      -> 4 bytes video temporal quality = long unsigned value (0 to 1024)
      -> 4 bytes video spatial quality = long unsigned value (0 to 1024)
      - some quality values are lossless = 1024 ; maximum = 1023 ; high = 768
      - some quality values are normal = 512 ; low = 256 ; minimum = 0
      -> 4 bytes video frame pixel size = short unsigned width + short unsigned height
      -> 8 bytes video resolution = long fixed point horizontal + long fixed point vertical
      -> 4 bytes video data size = long value set to zero
      -> 2 bytes video frame count = short unsigned total (set to 1)
      -> 1 byte video encoding name string length = byte unsigned length
      -> 31 bytes video encoder name string
      -> NOTE: if video encoder name string < 31 chars then pad with zeros
      -> 2 bytes video pixel depth = short unsigned bit depth
      - colors are 1 (Monochrome), 2 (4), 4 (16), 8 (256), 16 (1000s), 24 (Ms), 32 (Ms+A)
      - grays are 33 (B/W), 34 (4), 36 (16), 40(256)
      -> 2 bytes video color table id = short integer value (no table = -1)
      -> optional color table data if above set to 0 (see color table atom below for layout)

    * 8+ bytes optional video gamma level atom = long unsigned offset + long ASCII text string
    'gama'
      -> 4 bytes decimal level = long fixed point level

    * 8+ bytes optional video field order atom = long unsigned offset + long ASCII text string
    'fiel'
      -> 2 bytes field count/order = byte integer total + byte integer order

    * 8+ bytes optional video m-jpeg quantize table atom = long unsigned offset + long ASCII text
    string 'mjqt'
      -> quantization table = hex dump

    * 8+ bytes optional video m-jpeg huffman table atom = long unsigned offset + long ASCII text
    string 'mjht'
      -> huffman table = hex dump

      -> 2 bytes audio channels = short unsigned count (mono = 1; stereo = 2)
      -> 2 bytes audio sample size = short unsigned value (8 or 16)
      -> 2 bytes audio compression id = short integer value
      - default = 0; audio size before decompression = 1; VBR = -2
      -> 2 bytes audio packet size = short value set to zero
      -> 4 bytes audio sample rate = long unsigned fixed point rate

```

```

-> optional extended atom-styled audio sample description for QT decompressor
  * 8+ bytes optional audio wave header chunk atom = long unsigned offset + long ASCII text
string 'wave'
  -> RIFF Waveform audio header 'fmt ' chunk contents = hex dump

* 8+ bytes time to sample atom = long unsigned offset + long ASCII text string 'stts'
-> 4 bytes version/flags = byte hex version + 24-bit hex flags (current = 0)
-> 4 bytes number of times = long unsigned total
-> 8+ bytes time per sample = long unsigned count + long unsigned duration

* 8+ bytes optional sync sample atom = long unsigned offset + long ASCII text string 'stss'
-> 4 bytes version/flags = byte hex version + 24-bit hex flags (current = 0)
-> 4 bytes number of key frames = long unsigned total
-> 4+ bytes key frame location = long unsigned sample number

* 8+ bytes sample to chunk atom = long unsigned offset + long ASCII text string 'stsc'
-> 4 bytes version/flags = byte hex version + 24-bit hex flags (current = 0)
-> 4 bytes number of chunks = long unsigned total
-> 8+ bytes samples per chunk = long unsigned first chunk + long unsigned # of samples
-> 4+ bytes samples description id = long unsigned description number

* 8+ bytes sample size atom = long unsigned offset + long ASCII text string 'stsz'
-> 4 bytes version/flags = byte hex version + 24-bit hex flags (current = 0)
-> 4 bytes samples size for all = 32-bit integer size (different sizes = 0)
-> 4 bytes number of sample sizes = long unsigned total
-> 4+ bytes sample byte sizes = long unsigned byte values

* 8+ bytes chunk offset atom = long unsigned offset + long ASCII text string 'stco'
-> 4 bytes version/flags = byte hex version + 24-bit hex flags (current = 0)
-> 4 bytes number of offsets = long unsigned total
-> 4+ bytes chunk offsets = long unsigned byte values

* 8+ bytes larger chunk offset atom = long unsigned offset + long ASCII text string 'co64'
-> 4 bytes version/flags = byte hex version + 24-bit hex flags (current = 0)
-> 4 bytes number of offsets = long unsigned total
-> 8+ bytes larger chunk offsets = 64-bit unsigned byte values

* 8+ bytes user data (any custom info) atom = long unsigned offset + long ASCII text string 'udta'
- see below (mainly just for custom track names and annotations related to tracks)

* 8+ bytes user data (any custom info) atom = long unsigned offset + long ASCII text string 'udta'

* 8+ bytes optional annotation atoms = long unsigned offset + 0xA9 + 24-bit ASCII text string
- atom types are full name = 'nam'; copyright = 'cpy'; disclaimer = 'dis'
- atom types are movie description = 'inf' or 'des'; comment = 'cmt'; warning = 'wrn'
- atom types are content created date = 'day'; movie edit dates = 'ed1' to 'ed9'
- atom types are original format = 'fmt'; original source = 'src'; host computer = 'hst'
- atom types are make = 'mak'; model = 'mod'; product = 'PRD'; software = 'swr'
- atom types are playback requirements = 'req'; director = 'dir'; producer = 'prd'
- atom types are performers = 'prf'; writer = 'wrt'; author = 'aut'
- atom types are artist = 'ART'; track = 'trk'; album = 'alb'; composer = 'com'
- atom types are genre = 'gen'; original artist = 'ope'; net url = 'url'; encoder = 'enc'
-> 2 bytes string length = short unsigned length
-> 2 bytes mac language = short unsigned language value (english = 0)
-> annotation string = ASCII text dump

* 8+ bytes optional track/movie name atom = long unsigned offset + long ASCII text string 'name'
-> track name string data = ASCII text string

* 8+ bytes optional controller type atom = long unsigned offset + long ASCII text string 'ctyp'
-> 4 bytes controller name = long ASCII text string
- some controller names are 0x00 + 'std' for linear movie ; 'none' for no controls

* 8+ bytes optional window location atom = long unsigned offset + long ASCII text string 'WLOC'
-> 4 bytes coordinates = short integer x value + short integer y value

* 8+ bytes optional looping atom = long unsigned offset + long ASCII text string 'LOOP'
-> 4 bytes looping mode = long integer value
- mode types are normal = 0 ; palindromic = 1

* 8+ bytes optional play selection only atom = long unsigned offset + long ASCII text string 'sel0'
-> 1 byte play select mode = byte integer value
- mode types are disabled = 0 ; enabled = 1

* 8+ bytes optional play all frames atom = long unsigned offset + long ASCII text string 'AllF'
-> 1 byte play all mode = byte integer value
- mode types are disabled = 0 ; enabled = 1

* 8+ bytes optional auto play atom = long unsigned offset + long ASCII text string 'play'
-> 1 byte auto play mode = byte integer value
- mode types are disabled = 0 ; enabled = 1

* 8+ bytes optional no saving atom = long unsigned offset + long ASCII text string 'nsav'
-> 4 bytes disable mode = long hex value
- mode types are savable = 0 ; protected = 1

```

```

* 8+ bytes optional plugin parameter atom = long unsigned offset + long ASCII text string 'plug'
-> embed attribute string = ASCII text dump

* 8+ bytes optional qt player present atom = long unsigned offset + long ASCII text string 'ptv '
-> 2 bytes present mode = short integer value
  - mode types are normal = 0 ; double = 1 ; half = 2 ; full = 3 ; current = 4
-> 4 bytes flags = long value set to 0
-> 1 byte slideshow present mode = byte integer value
  - mode types are disabled = 0 ; enabled = 1
-> 1 byte auto present mode = byte integer value
  - mode types are disabled = 0 ; enabled = 1

* 8+ bytes optional chapter mode atom = long unsigned offset + long ASCII text string 'chmd'
-> 1 byte chapter mode = byte integer value
  - mode types are all = 0 ; clip = 1

* 8+ bytes optional qt player adjust video atom = long unsigned offset + long ASCII text string 'TVAD'
-> 4 bytes video luma black level (brightness) = long integer value
  - values are darker = negatives ; normal = 0 ; lighter = positives
-> 4 bytes video luma white level (contrast) = long integer value
  - values are darker = negatives ; normal = 0 ; lighter = positives
-> 4 bytes video chroma balance (tint) = long integer value
  - values are more chroma red = negatives ; normal = 0 ; more chroma blue = positives
-> 4 bytes video chroma level (color) = long integer value
  - values are reduce color = negatives ; normal = 0 ; increase color = positives

* 8+ bytes optional app quit when done atom = long unsigned offset + long ASCII text string 'aqui'
-> 1 byte app quit mode = byte integer value
  - mode types are disabled = 0 ; enabled = 1

* 8+ bytes optional adjust volume on import atom = long unsigned offset + long ASCII text string 'volu'
-> 2 bytes volume = short unsigned level (100% = 256)

* 8+ bytes optional custom atom = long unsigned offset + long ASCII text string
  - some atom types are Adobe Premiere string = '@PRM' ; Adobe Premiere QT string = '@PRQ'
  - atom type for Media Cleaner Pro string = 'MCPS'
  - remember custom atoms can only use uppercase text characters

-> 4 bytes compatibility utda end = long value set to zero

* 8+ bytes color mapping table atom = long unsigned offset + long ASCII text string 'ctab'
-> 4 bytes seed value = long value set to zero
-> 2 bytes flags = short hex value (set to 0x8000)
-> 2 bytes size/number of colors = short unsigned index total (one color = 0)
-> 2+ bytes array value = short unsigned index value
-> 6+ bytes color values = 3 * short unsigned RGB values

*****
* QuickTime 5 Install/Archive Package Format *
*****

- values use big endian (network) byte order
- general terms: integer = signed value
- general values: byte = 8-bit value; short/word = 16-bit value; long = 32-bit value
- octal values: base-8 long unsigned values (values containing 8 and 9 are invalid)
- atom offsets: values relative to atoms only and are used to skip to the next atom
- long mac file version: byte hex version + byte hex revision + byte hex revision stage + byte hex non-final
number
- mac version revision stages: development = 0x20 ; alpha = 0x40 ; beta = 0x60 ; release = 0x80

FILE INFO

Suffix = ".pkg"; Mac OS Type = "pack"; Mac OS Creator = "qtup"; MIME="application/x-quicktime-install-
package"

Standard single fork binary file that only uses a resource fork on HFS/HFS+ volumes to store mac
specific file info.

Unknown atoms can be safely skipped over, atoms can be in any order, all lowercase long ASCII text strings
used for atom names/types are reserved for use by Apple and packages can contain multiple file atoms.

The format is limited to a total size of 2^32 bytes (< 4.3 GB).

FILE CONTAINER/HEADER

* 8+ bytes package container atom = long unsigned offset + long ASCII text string 'pckg'

* 8+ bytes optional package file name atom = long unsigned offset + long ASCII text string 'pnam'
-> name of this file = ASCII text string

* 8+ bytes optional package title name atom = long unsigned offset + long ASCII text string 'punm'
-> display name = ASCII text string

* 8+ bytes optional target os atom = long unsigned offset + long ASCII text string 'tos '

```

```

-> 3 bytes os type = 24-bit ASCII text string
  - OS types are Mac OS = 'mac'; Mac OS X = 'osx'; MS 32-bit Windows = 'win'

* 8+ bytes optional target language atom = long unsigned offset + long ASCII text string 'lang'
-> 2 bytes language code = short ASCII text string
  - some language codes are English - North American = 'us'; English - British = 'uk'
  - some language codes are German = 'de'; French = 'fr'; Italian = 'it'

* 8+ bytes optional package task atom = long unsigned offset + long ASCII text string 'pact'
-> 4 bytes system code = long ASCII text string
  - a system code for Restart = 'rsrt'

* 8+ bytes optional license atom = long unsigned offset + long ASCII text string 'lice'

* 8+ bytes optional license maintainer atom = long unsigned offset + long ASCII text string 'lcmn'
-> 4 bytes company code = long ASCII text string
  - a company code for Apple = 'appl'

* 8+ bytes optional license ID atom = long unsigned offset + long ASCII text string 'lcid'
-> 4 bytes unique ID = long integer

* 8+ bytes optional license language atom = long unsigned offset + long ASCII text string 'lang'
-> 2 bytes language code = short ASCII text string
  - some language codes are English - North American = 'us'; English - British = 'uk'
  - some language codes are German = 'de'; French = 'fr'; Italian = 'it'

* 8+ bytes optional license agree atom = long unsigned offset + long ASCII text string 'lcag'
-> button name = ASCII text string

* 8+ bytes optional license disagree atom = long unsigned offset + long ASCII text string 'lcdg'
-> button name = ASCII text string

* 8+ bytes license text atom = long unsigned offset + long ASCII text string 'lctx'
-> user viewable text = ASCII text dump

* 8+ bytes file atom = long unsigned offset + long ASCII text string 'file'

* 8+ bytes compression atom = long unsigned offset + long ASCII text string 'comp'
-> 4 bytes compression code = long ASCII text string
  - a compression code for Deflate = 'zlib'

* 8+ bytes optional version atom = long unsigned offset + long ASCII text string 'vers'
-> 4 bytes mac file version = long hex value (see above)

* 8+ bytes optional file modified date atom = long unsigned offset + long ASCII text string 'fmdt'
-> 4 bytes mac file date = long unsigned value in seconds since beginning 1904 to 2040

* 8+ bytes displayed file name atom = long unsigned offset + long ASCII text string 'fnam'
-> displayed file name = ASCII text string

* 8+ bytes unpacked file name atom = long unsigned offset + long ASCII text string 'unam'
-> actual file name = ASCII text string

* 8+ bytes file creator atom = long unsigned offset + long ASCII text string 'fcrf'
-> 4 bytes mac os creator code = long ASCII text string

* 8+ bytes file type atom = long unsigned offset + long ASCII text string 'ftyp'
-> 4 bytes mac os type code = long ASCII text string

* 8+ bytes location atom = long unsigned offset + long ASCII text string 'loca'
-> 4 bytes install location code = long ASCII text string
  - some location codes are qt app directory = 0x7174C420 ; system directory = 'macs'
  - some location codes are system init directory = 'extn' ; qt system directory = 'qtex'
  - some location codes are web browser directories = 0xC46E6574 ; system cdev directory = 'ctrl'
  - some location codes are control strip folder = 'sdev' ; loc. manager folder = 'walk'
  - some location codes are mac os runtime java folder = 0x6D726AC4 ; os x qt system directory = 'qt3c'

* 8+ bytes installer action atom = long unsigned offset + long ASCII text string 'inac'
-> 4 bytes installer action code = long ASCII text string
  - some installer action codes are plain file = 'file' ; system add-on = 'init'
  - some installer action codes for system component = 'thng'

* 8+ bytes data fork atom = long unsigned offset + long ASCII text string 'dfrk'
-> 4 bytes uncompressed size = long unsigned value
-> compressed data = hex dump

* 8+ bytes mac resource fork atom = long unsigned offset + long ASCII text string 'rfrk'
-> 4 bytes uncompressed size = long unsigned value
-> compressed data = hex dump

```

```

*****
* QuickTime 3 Image Format *
*****

```

- values use big endian (network) byte order
- general terms: integer = signed value
- general values: byte = 8-bit value; short/word = 16-bit value; long = 32-bit value
- fixed point values: value made up of an integer for whole numbers and an unsigned value for the decimal
- octal values: base-8 long unsigned values (values containing 8 and 9 are invalid)
- atom offsets: values relative to atoms only and are used to skip to the next atom
- long mac file version: byte hex version + byte hex revision + byte hex revision stage + byte hex non-final number
- mac version revision stages: development = 0x20 ; alpha = 0x40 ; beta = 0x60 ; release = 0x80

FILE INFO

Suffixes = ".qtif", ".qti"; Mac OS Type = "qtif"; Mac OS Creator = "ogle"; MIME="image/x-quicktime"

Standard single fork binary file that only uses a resource fork on HFS/HFS+ volumes to store mac specific file info and image previews.

Unknown atoms can be safely skipped over, atoms can be in any order and all lowercase long ASCII text strings used for atom names/types are reserved for use by Apple.

All atoms are limited to a size of 2³² bytes (< 4.3 GB).

FILE IMAGE DATA

- * 8+ bytes image data atom = long unsigned offset + long ASCII text string 'idat'
- > image data = hex dump
- JPEG YUV scans are stored as standard JFIFs

OPTIONAL/UNOFFICIAL FILE PREVIEW DATA

Note: the image preview can be stored in the resource fork as type 'pnot'; ID any on the Macintosh platform. But could be stored in a single file fork. The advantage of using another file fork is that the preview image data can be updated more easily.

- * 8+ bytes optional preview location, not image data atom = long unsigned offset + long ASCII text string 'pnot'
- > 4 bytes mac modified file date = long unsigned value in seconds since beginning 1904 to 2040
- > 2 bytes version = short integer value (none = 0)
- > 4 bytes resource/atom type = long ASCII text string
- types are separate preview = 'PICT' ; use image data = 'moov'
- > 2 bytes resource/atom id = short integer value
- IDs are separate preview = id value ; use image data = -1
- * 8+ bytes optional preview data for single fork atom = long unsigned offset + long ASCII text string 'PICT'
- > image data = quickdraw picture resource format hex dump

FILE IMAGE HEADER

Note: the header should be stored at the beginning of the file. The following is the same layout as used in the QuickTime movie video sample description and QuickDraw PICT format description for QuickTime encoded images, but is in an image description atom container.

- * 8+ bytes image description atom = long unsigned offset + long ASCII text string 'idsc'
- > 4 bytes image description size = long unsigned length
- > 4 bytes subtype/compressor name = long ASCII text string (eg. Photo - JPEG = 'jpeg')
- > 8 bytes reserved = 64-bit null space
- > 4 bytes version = long fixed point value (none = 0.0)
- > 4 bytes manufacturer name = long ASCII text string (eg. Apple = 'appl')
- > 4 bytes image temporal quality = long unsigned value (0 to 1024; generally = 0)
- > 4 bytes image spatial quality = long unsigned value (0 to 1024)
- some quality values are lossless = 1024 ; maximum = 1023 ; high = 768
- some quality values are normal = 512 ; low = 256 ; minimum = 0
- > 4 bytes image frame pixel size = short unsigned width + short unsigned height
- > 8 bytes image resolution = long fixed point horizontal + long fixed point vertical
- > 4 bytes image data size = long unsigned byte size (unknown = 0)
- > 2 bytes image frame count = short unsigned total (default = 1)
- > 1 byte image encoding name string length = byte unsigned length
- > 31 bytes image encoder name text string
- > NOTE: if image encoder name string < 31 chars then pad with zeros
- > 2 bytes image pixel depth = short unsigned value
- colors are 1 (Monochrome), 2 (4), 4 (16), 8 (256), 16 (1000s), 24 (Ms), 32 (Ms+A)
- grays are 33 (B/W), 34 (4), 36 (16), 40(256)
- > 2 bytes image color table id = short integer value (no table = -1)
- > optional color table values if above set to 0 (see movie color table atom for layout)
- > optional atom-styled extended image description for QT decompressor

OPTIONAL FILE META INFO

- * 8+ bytes optional image meta atom = long unsigned offset + long ASCII text string 'meta'
- * 8+ bytes optional annotation atoms = long unsigned offset + 0xA9 + 24-bit ASCII text string
- atom types are full name = 'nam'; copyright = 'cpy'; disclaimer = 'dis'
- atom types are image description = 'inf' or 'des'; comment = 'cmt'; warning = 'wrn'
- atom types are content created date = 'day'; image edit dates = 'ed1' to 'ed9'
- atom types are original format = 'fmt'; original source = 'src'; host computer = 'hst'
- atom types are make = 'mak'; model = 'mod'; product = 'PRD'; software = 'swr'
- atom types are viewer requirements = 'req'; director = 'dir'; producer = 'prd'
- atom types are performers = 'prf'; writer = 'wrt'; author = 'aut'
- atom types are artist = 'ART'; track = 'trk'; album = 'alb'; composer = 'com'

```
- atom types are genre = 'gen'; original artist = 'ope'; net url = 'url'; encoder = 'enc'  
-> 2 bytes string length = short unsigned length  
-> 2 bytes mac language = short unsigned language value (english = 0)  
-> annotation string = ASCII text dump
```