Becca Bender
Digital Preservation
CINE-GT 1807
12/15/17

Too Many Quicktimes! (and other related confusions): Thoughts from an Emerging Audiovisual Archivist

I am going on the record to say: **There are simply too many QuickTimes.** One can accurately refer to no less than two distinct media players, one codec, and two different file formats as "QuickTime," but without a considerable amount of knowledge about audiovisual digital architecture, it can be quite difficult to comprehend the nature of each of these "QuickTimes" and to understand how they relate to the others of their namesake. In an attempt to overcome my own confusion with respect to QuickTime, (and perhaps shed light for others) this paper will seek to provide a basic history and foundation for each incarnation of "QuickTime," while keeping an eye toward the challenges for digital preservation when terminology lacks specificity. In light of this latter conundrum, I will try to avoid ambiguous use of the word "format" (one of the most problematic in the realm of digital AV clarity), and stick to the words "codec" and "container" when referring to encodings and file formats.

A current search for the word "QuickTime" on Apple's main homepage predominantly brings up links related to QuickTime's player(s),[1] the primary context through which a mainstream user will understand the brand. In contrast however, a search through Apple's library of "Guides and Sample Code" for developers, will lead to the following definition as stated in the "QuickTime Overview:"

> QuickTime is a complete multimedia architecture, not just a media player. It supports creating, producing, and delivering a broad variety

[1] "Search Apple.com," *Apple,* https://www.apple.com/us/search/quicktime?src=globalnav (accessed December 13, 2017). Search results include both QuickTime Player 7 and QuickTime Player X.

> of media. QuickTime provides end-to-end support for the entire process: capturing media in real time; synthesizing media programmatically; importing and exporting existing media; editing and compositing; compression, delivery, and user playback.[2]

This first sentence, in which Apple informs the reader that QuickTime is "not just a media player," makes clear that the company understands it has a clarity issue on its hands. Unfortunately, in the 26 years since QuickTime was released, Apple seems to have done very little to differentiate the various pieces of QuickTime, with the exception of assigning versioning numbers for the players.

QuickTime, with all its component parts, was first introduced by Apple in 1991. QuickTime 1.0 was a media player that offered four codecs for compression: Video, JPEG, Animation, and Graphics, as well as the ability to export a QuickTime Movie file, also, at the time, referred to simply as a "Movie." As evidenced by the names of the codecs, the decision as to how a user encoded a Movie was generally based on the type of content. One early write-up of QuickTime 1.0 stated: "For live action, the currently recommended QuickTime choice is the default Video CODEC which saves images at 16 bits per pixel."[3] This "Video" codec, (full name "Apple Video"), it seems was often referred to, like most everything else in the architecture, simply as "QuickTime." Eventually, in what I can only assume was an attempt by users to differentiate the codec from other aspects of QuickTime, and to avoid use of the generic word "video," "Apple Video" came to be known commonly by its FourCC identifier RPZA, or its nickname, "Road Pizza." Placed in historical context, Apple's initial failure to semantically

---

[2] "Guides and Sample Code: QuickTime Overview," *Apple Developer,* https://developer.apple.com/library/content/documentation/QuickTime/RM/Fundamentals/QTOverview/QTOverview_Document/QuickTimeOverview.html#//apple_ref/doc/uid/TP30000992-CH1g-QuickTimeOverview (accessed December 13, 2015.

[3] Judson Rosebush, "Adventures in QuickTime," *CD-ROM Professional,* November 1992, 17.

distinguish QuickTime's most commonly used codec from its container makes some sense when one considers that "back in the 90s, people didn't define containers and encodings as being separate from each other."[4] Furthermore, Apple was clearly trying to establish and market QuickTime's brand, and perhaps the idea was just to say the word as much as possible. Interestingly, Apple's branded "Apple Animation," codec, one that remained in use long past "Road Pizza," became colloquially known as "Quicktime Animation." So in that case, even though officially QuickTime Player identified such a stream as "Apple Animation," users themselves inserted the word "QuickTime" into the codec name.[5]

Breaking away from the specifics of QuickTime, I'd like to take a moment for a general consideration of how codec and container identity is determined, and the resulting impact on preservation. In order to properly preserve digital objects, one must not only safeguard the literal bits, but also practice logical preservation, "the process of preserving file accessibility and ensuring that they are still understandable and readable, regardless of the evolving technologies.[6] One of the central requirements for the logical preservation of audiovisual digital objects is to know the codecs and containers of one's files, which in turn set the parameters for the software and computing environments needed to render those files. Individual archives have varying workflows with respect to how they gather and document this technical metadata, and it's not hard to imagine scenarios in which the metadata reporting tools used by an organization would change over time, or even that the individual tools themselves might refine their terminology

[4] Dave Rice, Interview with the author, November 22, 2017.
[5] This observation is based on the author's own experience, having worked in media production for many years.
[6] "Bit Preservation and Logical Preservation - Wiki.Dca-Project.Eu," https://wiki.dca-project.eu/wiki/Bit_preservation_and_logical_preservation (accessed December 13, 2017).

with evolving releases. Using a file encoded with QuickTime Animation (contained in a QuickTime file format) as an example, one could wind up with the following video codec results depending on the software:

Program	Function/View	Descriptor Type	Value
QuickTime Player 7	Inspector (Command-I)	Format	Apple Animation
MediaInfo GUI	Text View	Video: Format Video: Format/Info	RLE Run-length encoding
VLC Player 2.2.6	Media Information (Command-I)	Codec	Apple QuickTime RLE Video
Mac Finder	Get Info	Codecs	Animation

Here is the same test performed with an Apple Video “Road Pizza” encoding wrapped in a QuickTime file format:

Program	Function/View	Descriptor Type	Value
QuickTime Player 7	Inspector (Command-I)	Format	Apple Video
MediaInfo GUI	Text View	Video: Format Video: Codec ID	Road Pizza rpza
VLC Player 2.2.6	Media Information (Command-I)	Codec	Apple Video (rpza)
Mac Finder	Get Info	Codecs	Video

And finally, an Apple Video "Road Pizza" encoding wrapped in the Audio Video Interleave (AVI) file format:

Program	Function/View	Descriptor Type	Value
QuickTime Player 7	Inspector (Command-I)	Format	Apple Video
MediaInfo GUI	Text View	Video: Format Video: Codec ID	azpr azpr
VLC Player 2.2.6	Media Information (Command-I)	Codec	Apple Video (azpr)
Mac Finder	Get Info	NA	NA

None of the programs yield the exact same result for any of the three files, nor do they necessarily even use consistent terminology within their own reporting (e.g. VLC describes Apple Animation and Apple Video very differently even though Apple was consistent in how it named them). Of course it doesn't take much googling to figure out that all of these programs are talking about the same types of encodings, but an archivist who doesn't specialize in audiovisual materials (and perhaps even one who does) would likely have no inclination to perform that google search. Nor would they know, for example, that an optical disc marked "QuickTime Animations" is referring to the same thing that MediaInfo calls "RLE" or VLC calls "Apple QuickTime RLE Video." Ultimately, this variability with how a codec might be identified, and therefore recorded, acts in direct opposition to an archive's ability to efficiently practice logical preservation.

As confusing as it may be to have two (or more) different names for the same thing, it's far worse to have the *same* name for two different things! Sadly, for all intents and purposes, this

is the situation with respect to the QuickTime File Format. As stated earlier, when QuickTime was first released in 1991, the architecture included a container format called "QuickTime Movie." Although Apple's developer guide "QuickTime Overview" states: "In casual use, a QuickTime movie file is sometimes simply called a movie,"[7] I would argue that more to the point, a QuickTime movie file is generally simply called a "QuickTime." This would already be confusing given that there are also QuickTime codecs and QuickTime software, but it becomes absolutely head-spinning when one realizes that the QuickTime container originally designed by Apple, and used through the early 2000s, (referred to here going forward as QTFF Classic) is different than the "QuickTime" container that we generally use today. And in point of fact, this "current" QuickTime file format is not *really* QuickTime at all, but MPEG-4 Part 12 (aka ISO Base Media File Format) with a QuickTime profile.

In order to unpack that last sentence, I need to switch gears and talk about MPEG-4 for a moment. MPEG-4 is an evolving group of standards to define the "coding of audio-visual objects, "[8] under the direction of the ISO/IEC Motion Picture Experts Group. When the development of MPEG-4 began in 1998, a joint proposal from six of the biggest names in computing (Apple, IBM, Netscape, Oracle, Silicon Graphics and Sun Microsystems) put forward the idea that the standard should be built using "Apple's QuickTime File Format [QTFF Classic] as the starting point for the development of a unified digital media storage format."[9] The Motion

[7] "Guides and Sample Code: QuickTime Overview," *Apple Developer,* https://developer.apple.com/library/content/documentation/QuickTime/RM/Fundamentals/QTOverview/QTOverview_Document/QuickTimeOverview.html#//apple_ref/doc/uid/TP30000992-CH1g-QuickTimeOverview (accessed December 13, 2015).
[8] "Standards: MPEG-4," *The Motion Picture Experts Group*, https://mpeg.chiariglione.org/standards/mpeg-4 (accessed December 13, 2015).
[9] "ISO Adopts QuickTime File Format as Starting Point for Developing Key Component of MPEG-4 Specification," *PR Newswire*, February 11, 1998, 1.

Picture Experts Group agreed with the consortium's proposal, stating that "By the adoption of the QuickTime file format as the starting point for an MPEG-4 standard, users are assured that all digital media content can be authored in a common file format which also supports real-time video and audio streaming."[10]

As stated above, MPEG-4 (aka ISO/IEC 14496) is a *group* of standards, at present comprised of 33 parts. Those various parts define several different aspects of the overall standard, but the one of central concern to this paper is MPEG-4 Part 12 (ISO/IEC 14496-12) which specifies the ISO base media file format (IBMFF). It is IBMFF that is directly built upon the structure of QTFF Classic, but the key thing to take away is that unlike QTFF Classic which I would call a "standalone" container format, IBMFF is a generic file format upon which *other* file formats within the MPEG-4 family are built, (such as MP4, 3GP, and Motion JPEG 2000). So unlike QTFF Classic which almost always used the file extension .mov (although to complicate matters it can also use .qt), formats that are built on IBMFF have many different file extensions depending on their "profile,[11]" and are almost always known by their "brand" names as opposed to being called IBMFFs. This would all be well and good, except that the current version of the QuickTime container (QTFF Current) is *also* built on IBMFF, and its brand name already refers to a *different* container, as well as multiple codecs and media players! Furthermore, like QTFF Classic, QTFF Current uses the extension .mov which means that there's also no visual cue in the file name to alert the user that they have one QuickTime file format over the other.

[10] Ibid., 2.

[11] A helpful list of these file extensions can be generated by running MediaInfo with a "full" flag (-f) in the command line: "Format/Extensions usually used: mov mp4 m4v m4a m4b m4p 3ga 3gpa 3gpp 3gp 3gpp2 3g2 k3g jpm jpx mqv ismv isma ismt f4a f4b f4v."

This brings us back around to the use of metadata reporting tools and the confusion that they can inadvertently cause. Starting with the basic "Get Info" option through the Mac Finder, (the way I would have "inspected" a file prior to my training in audiovisual archiving), one receives the exact same result irrespective of whether a file is QTFF Classic or QTFF Current: both files will report that the "Kind" is "QuickTime Movie." At the time I began this project, I had assumed that the "[Date] Created" reported by "Get Info" would at least allow the user to *deduce* which type of QTFF they had (pre mid-2000s vs post mid-2000s), but in the course of my research I have been the recipient of two recently created QTFF Classic files,[12] so there goes that theory. Switching then to MediaInfo, a more granular reporting tool, I find that things can get simultaneously more clear and more confusing. Fortunately in the case of QTFF Classic, MediaInfo is wonderfully explicit. In the "General" section of the GUI report, the "Format" value is defined as "QuickTime" and a "Format/Info" field is included with the value "Original Apple specifications." Perfect! Very clear! The manner in which MediaInfo reports on QTFF Current files, however, is not so straightforward. To begin, the "Format" is defined as "MPEG-4," and instead of having a "Format/Info" field like with QTFF Classic, there is a field called "Format Profile" for which the value is "QuickTime." Understanding what I do now about IBMFF, I see that MediaInfo is trying to tell its user that the container is a QuickTime profile of MPEG-4 Part 12, a type of ISO base media file format. Or, as MediaInfo developer Jerome

[12] The first file is one that I made myself when digitizing a Umatic tape using a BlackMagic Design Video Assist, a standalone capture card that does not require a computer or updates. When using that hardware's Apple ProRes encoding setting, the file was automatically wrapped in QTFF Classic. The second QTFF Classic file was also encoded with Apple ProRes, and was sent to me by the Library of Congress, transcoded from a JPEG2000 encoding wrapped in MXF, that had been created from a film scan just weeks before.

Martinez put it in a online forum: "new QTFF is now a 'flavour' of MP4."[13] Prior to completing the research for this paper however, I was completely confused by MediaInfo's reporting on QTFF Current, and if I had been asked to record the container format, I would have inaccurately written down MP4, a mistake that I imagine is not made infrequently.

My understanding from conversations with digital archivist Dave Rice, is that MediaInfo is trying to simplify matters by shortening "MPEG-4 Part 12" to "MPEG-4," and foregoing use of the lengthy phrase "ISO base media file format." But at the same time that MediaInfo chooses to be generic with the "Format" field of a QTFF Current file, it is quite specific when it comes to the "Codec ID" field two lines down, which is defined as "qt 2005.03 /(qt)." Unfortunately the field name, "Codec ID", is a complete misnomer since the "General" section of MediaInfo refers to the container, not the codec, but the value here is actually helpful information when you know what you're looking at: the file type designation for a certain type of QuickTime atom, i.e., another clue that the file should be described at least in part as "QuickTime."

I don't mean to point out these MediaInfo reporting concerns to discredit the software which I think is an invaluable tool. Rather, I use these examples to demonstrate how variations in terminology and granularity can be extremely challenging when one does not have a full scope of knowledge. Returning to the idea of logical preservation, knowing whether one has QTFF Classic files, QTFF Current files (IBMFF with a QuickTime profile) or actual MP4 files (MPEG-4 Part 1 or Part 14) could be critical for understanding the files' renderability dependencies in the future.

[13] "MediaInfo / Discussion / Open Discussion: ProRes Info," April 17, 2015, *Sourceforge*, https://sourceforge.net/p/mediainfo/discussion/297609/thread/87852be6/?limit=25#c9c3/e2f9 (accessed December 15, 2017).

Even today, the renderability of certain files can depend on *another* QuickTime distinction, that of QuickTime Player 7 versus QuickTime Player X. QuickTime Player X was released in 2009, and like Final Cut X, the release marked a major break in Apple's relationship with media makers and media archivists. By removing characteristics such as the frame counter, Command-J property windows, clear frame boundaries and 2x speed audio playback, I would argue that QuickTime Player X is simply less useful for users who are trying to closely inspect media files. But equally important, QuickTime Player X will simply not play certain older media files, such as those encoded with "Road Pizza" or Apple Animation, that QuickTime Player 7 is still capable of rendering. Furthermore, QuickTime 7 has already been completely deprecated from Windows operating systems, and Mac users are weary that it's not long for OS X either. So once again, if we consider the less experienced archivist, what happens when that person is told, for example: "you'll be able to open it in QuickTime." What does that sentence even mean? There's no reason to think that someone would expect there to be a QuickTime Player *other* than the one that came on their Apple computer. So what is that person to think when they have a QuickTime file, encoded with QuickTime, that cannot be played on QuickTime?

Bibliography/Webography

"Bit Preservation and Logical Preservation - Wiki.Dca-Project.Eu." Accessed December 15, 2017. https://wiki.dca-project.eu/wiki/Bit_preservation_and_logical_preservation.

Farrell, John, "Whatever Happened to QuickTime?" *Streaming Media*, October/November 2007.

"Final Cut Pro 7 User Manual: The QuickTime Movie File Format." Accessed November 4, 2017. https://documentation.apple.com/en/finalcutpro/usermanual/index.html#chapter=103%26section=3%26tasks=true.

Green, Doug and Green, Denise, "Apple's QuickTime Reaches for Multimedia Future." *InfoWorld*. July 8, 1991.

Hone, Robert, *QuickTime: Making Movies With Your Macintosh.* Rocklin, CA: Prima Publishing, 1995.

"Installing QuickTime Player 7 on Your Mac." Apple Support. Accessed November 19, 2017. https://support.apple.com/en-us/HT201288.

"ISO Adopts QuickTime File Format as Starting Point for Developing Key Component of MPEG-4 Specification." *PR Newswire,* February 11, 1998.

"ISO Base Media File Format | MPEG." Accessed November 19, 2017. https://mpeg.chiariglione.org/standards/mpeg-4/iso-base-media-file-format.

"ISO Base Media File Format." *Wikipedia*, November 10, 2017. https://en.wikipedia.org/w/index.php?title=ISO_base_media_file_format&oldid=80962939.

"MediaInfo / Discussion / Open Discussion:ProRes Info." Accessed December 15, 2017. https://sourceforge.net/p/mediainfo/discussion/297609/thread/87852be6/?limit=25#c9c3/e2.

"MPEG-4 | MPEG." Accessed December 15, 2017. https://mpeg.chiariglione.org/standards/mpeg-4.

Ozer, Jan, "Will MPEG-4 Fly?" *PC Magazine*, April 3, 2001.

"QTIF File Extension - What Is a .Qtif File and How Do I Open It?" Accessed November 5, 2017. https://fileinfo.com/extension/qtif.

"QuickTime Animation." *Wikipedia*, September 10, 2017. https://en.wikipedia.org/w/index.php?title=QuickTime_Animation&oldid=799864387.

"QuickTime Container - MultimediaWiki." Accessed November 19, 2017. https://wiki.multimedia.cx/index.php?title=QuickTime_container.

"QuickTime File Format." *Wikipedia*, September 10, 2017. https://en.wikipedia.org/w/index.php?title=QuickTime_File_Format&oldid=799864424.

"QuickTime File Format Specification," *Apple, Inc*., 2001.

"QuickTime Overview." Accessed December 15, 2017. https://developer.apple.com/library/content/documentation/QuickTime/RM/Fundamentals/QTOverview/QTOverview_Document/QuickTimeOverview.html#//apple_ref/doc/uid/TP30000992-CH1g-QuickTimeOverview.

Rice, Dave, Interview with the author, New York, November 22, 2017.

Rosebush, Judson. "Adventures in QuickTime." *CD-ROM Professional,* November 1992.

"Search Apple.com," Accessed December 13, 2017. https://www.apple.com/us/search/quicktime?src=globalnav

"Sustainability of Digital Formats Planning for the Library of Congress: QuickTime File Format." Web page, February 14, 2013. https://www.loc.gov/preservation/digital/formats/fdd/fdd000052.shtml.