Introduction to QuickTime File Format Specification

The QuickTime File Format (QTFF) is designed to accommodate the many kinds of data that need to be stored in order to work with digital multimedia. The QTFF is an ideal format for the exchange of digital media between devices, applications, and operating systems, because it can be used to describe almost any media structure.

The file format is object-oriented, consisting of a flexible collection of objects that is easily parsed and easily expanded. Unknown objects can simply be ignored or skipped, allowing considerable forward compatibility as new object types are introduced.

QuickTime itself provides a number of high-level functions that you can use to create and manipulate QuickTime files, without requiring you to understand the actual file format. These functions serve to insulate developers from the low-level details of operation. That said, not all kinds of QuickTime files can be created without the information presented here.

Important: The QuickTime File Format has been used as the basis of the MPEG-4 standard and the JPEG-2000 standard, developed by the International Organization for Standardization (ISO). Although these file types have similar structures and contain many functionally identical elements, they are distinct file types.

Warning: Do not use this specification to interpret a file that conforms to a different specification, however similar.

The *QuickTime File Format Specification* assumes that you are familiar with the basic concepts of digital video and audio, as well as with programming QuickTime and the QuickTime API. Note that this version of the document supersedes all previous versions of the *QuickTime File Format Specification*.

Organization of This Document

This document begins with an overview of QuickTime atoms, then presents the structure of the QuickTime file format in detail. This is followed by a series of code examples for manipulating a QuickTime file using the QuickTime API. Finally, a number of related topics are described in a series of appendixes. These include such topics as the handling of metadata when importing files into QuickTime, random access, and the QuickTime Image File format.

QuickTime files are described in general, rather than how they are supported on a specific computing platform or in a specific programming language. As a result, the file format information is presented in a tabular manner, rather than in coded data structures. Similarly, field names are presented in English rather than as programming language tags. Furthermore, to the extent possible, data types are described generically. For example, this book uses “32-bit signed integer” rather than “long” to define a 32-bit integer value.

QuickTime files are used to store QuickTime movies, as well as other data. If you are writing an application that parses QuickTime files, you should recognize that there may be non-movie data in the files.

QuickTime is a rich technology that continues to evolve as new practices and needs arise in audio/visual media. Because of this, certain elements of QuickTime technology may become deprecated over time. In order to preserve sufficient information about these legacy components for existing QuickTime files that include them, deprecated elements are marked with a note at the top of their section in this revision of the *QuickTime File Format Specification*.

Licensing Information

The *QuickTime File Format Specification* is provided for informational purposes. Apple may have patents, patent applications, trademarks, copyrights, or other intellectual property rights covering subject matter in this document. The furnishing of this document does not give you a license to any patents, trademarks, copyrights, or other intellectual property.

Important: For more information about licensing the QuickTime File Format, contact: Apple, Inc., Software Licensing Department, 12545 Riata Vista Circle, MS 198 3-SWL, Austin, TX 78727. Email Address: sw.license@apple.com

Special Fonts

All code listings, reserved words, and the names of actual data structures, constants, fields, parameters, and routines are shown in `code voice`.

Words that appear in **boldface** are key terms or concepts and are defined in the Glossary.

For More Information

For information about membership in Apple’s developer program and developer technical support, you should go to this URL:

Apple Developer

For information on registering signatures, file types, and other technical information, contact

Apple Developer Technical Support (DTS)

Apple, Inc.

1 Infinite Loop, M/S 303-2T

Cupertino, CA 95014