Technical Note QT03

Movies `LOOP' Atom and Friends

CONTENTS

This Technical Note discusses entertaining uses for QuickTime user data atoms, Apple defined and otherwise.

[Nov 01 1992]

Introduction

It is often desirable for an application to preserve the window position and looping state of a movie. This Technical Note describes the "Apple Sanctified" method of doing this using user data atoms.

User data atoms allow applications to store custom information which can be easily accessed using QuickTime Movie Toolbox calls. These user data atoms are text or data which can be associated and stored in any movie, track, or media. A reference to the list of user data atoms for each of these locations can be accessed with the following routines: `GetMovieUserData()`, `GetTrackUserData()`, and `GetMediaUserData()`. Once a reference to a list of user data atoms is obtained, an application can store, retrieve, and manage items in the list using the following routines: `GetNextUserDataType()`, `CountUserDataType()`, `AddUserData()`, `GetUserData()`, `RemoveUserData()`, `AddUserDataText()`, `GetUserDataText`, and `RemoveUserDataText()`. A complete description of these routines can be found in Inside Macintosh: QuickTime in the "Working With Movie User Data" section of chapter 2.

Back to top

User Data Types

Every user data atom carries a type identifier. This identifier is stored in a long integer, similar to a Resource Manager type. Apple has reserved all lowercase user data types values. Applications are free to use types values containing at least one uppercase letter.

In addition, since there is a differentiation between user data atoms which are handles to data and handles to text, Apple recommends that all text user data atoms begin with the (c) character (Option-G).

The following user data types containing text data are currently defined:

```
    '(c)cpy'    Copyright statement
    '(c)day'    Date the movie content was created
    '(c)dir'    Name of the movie's director
    '(c)ed1' to '(c)ed9'    Edit dates and descriptions
    '(c)fmt'    Indication of the movie format
     (computer-generated, digitized, and so on)
    '(c)inf'    Information about the movie
    '(c)nam'    Name of track or movie
    '(c)prd'    Name of the writer movie's producer
    '(c)prf'    Name of the performers
    '(c)req'    Special hardware or software requirements
    '(c)src'    Credits for those who provided the movie source content
    '(c)wrt'    Name of the movie's
```

Getting 'LOOP'-y

MoviePlayer(TM) has also defined two movie data atoms which are used to indicate looping and window location which applications can implement for compatibility with MoviePlayer(TM). They are:

```
'LOOP' If this 4 byte user data atom exists in the movie's user data list,
then looping is performed according to its value: 0 for normal looping and 1 for
palindrome looping

'WLOC' Handle to a point record indicating the last saved window position
```

Another variation on this which originated before MoviePlayer(TM) that applications should be aware of, is the following:

```
    'LOOP'    If this zero length data atom exists in the movie's user data list,
```

In summary, if a `'LOOP'` atom exists, then looping should be performed. If the returned data is a long integer of value 1, then palindrome looping should be performed. Normal looping should be performed if data returned is of zero length or if the returned data is a long integer of value 0.

Back to top

Sample Code

The following example demonstrates how to get looping information from a movie:

```
short loopInfo; //  0=no looping,1=normal looping,2=palindrome looping

    Handle       theLoop;
    Movie        theMovie;
    UserData     theUserData;

    loopInfo = 0;
    theLoop = NewHandle(0);
    theUserData = GetMovieUserData(theMovie);
    if (CountUserDataType(theUserData, 'LOOP')) {
        loopInfo = 1;
        GetUserData(theUserData, theLoop, 'LOOP', 1);
        if (GetHandleSize(theLoop))
            if ((** (long **) theLoop) == 1)
                loopInfo = 2;
    }
```

The following example demonstrates how to add a looping atom to a movie to indicate that user has selected looping:

```
    Handle       theLoop;
    Movie        theMovie;
    UserData     theUserData;
    short        theCount;

    theLoop = NewHandle(sizeof(long));
    (** (long **) theLoop) = 0;
    theUserData = GetMovieUserData(theMovie);
    theCount = CountUserDataType(theUserData, 'LOOP');
    while (theCount--)
        RemoveUserData(theUserData, 'LOOP', 1);
    AddUserData(theUserData, theLoop, 'LOOP');
```

The following example demonstrates how to remove a looping atom from a movie to indicate that looping is not selected:

```
Movie       theMovie;
    UserData     theUserData;
    short        theCount;

    theUserData = GetMovieUserData(theMovie);
    theCount = CountUserDataType(theUserData, 'LOOP');
    while (theCount--)
        RemoveUserData(theUserData, 'LOOP', 1);
```

Back to top

Inside the User Data Atom

Those of you who parse user data atoms directly by accessing the `'moov'` handle rather than with the appropriate movie toolbox calls, will notice a trailing long integer of value 0 after all user data atoms in the list. This is required for backward compatibility with QuickTime 1.0 which has a bug that requires the trailer. The size of the `'udta'` atom does reflect this extra trailing long integer. QuickTime 1.0 and future versions will automatically handle this when manipulating user data atoms with the movie toolbox calls.

Back to top

References

Inside Macintosh: QuickTime, Movie Toolbox Reference

Back to top

Downloadables

Acrobat version of this Note (K). Download
