

***TIME*LINE**



OpenTL File Format Specification

OpenTL 2.00
June 12, 2001

© TimeLine Vista, 1999, All rights Reserved

TimeLine Vista, Inc.
1755 La Costa Meadows Drive, Suite B
San Marcos, CA 92069
Tel (760) 761-4440
Fax (760) 761-4449

TimeLine OpenTL EDL Format

1 Project and Track Files

TimeLine OpenTL EDL projects will be stored in two parts:

1. A Project file that will contain project attributes and a list of track file names and track slot assignment info. Project files will always end with a ".tl" to indicate that it is an OpenTL EDL project file.
2. Track files (one for each track) that will contain track information as well as a list of events. The events will reference the actual sound media

In either case, these files (or messages) are terminated with the character '#'. This is equivalent to an end of file.

See Appendix 6.2 *Directory Names* for directory (folder) naming conventions

2 Properties and Text Conventions

Properties can be either simple or complex.

Attributes will be stored in the following format:

```
<TL_EDL> = ( <Property> )+
<Property> = ( <SimpleProperty> | <ComplexProperty> )

<SimpleProperty> = <FieldName> '(' <SimpleValue> ')'

<SimpleValue> = (Integer | "String" | float | 'char')

<ComplexProperty> = <FieldName> <ComplexValue>

<ComplexValue> = '{' <Property>* '}'
```

where <FieldName> must be exactly four characters long using only alpha numeric characters ('a' - 'z', 'A' - 'Z', '0' - '9').

For strings containing the letter '"' or '\', they must be escaped by replacing any occurrence of '"' with '\"', and any '\' with '\\'.

If a field is not understood, then it is to be ignored.

If an expected field is not present, then a default value will be used.

White space has no significance except in strings.

3 Definition of a DataSet Reference or FILE

References to files on volumes will be represented by the following definition of a Complex value.

```
FILE
'{ '
  <SimpleProperty>+
'}
```

The following simple properties are used for FILE:

```
VLTY
VLNM
DSNM
```

3.1 FILE FieldNames and descriptions

VLTY	Volume Type	String ("HFS", "HFS+", "FAT-16", "FAT-32")
VLNM	Volume Name	String (Described in the Appendix at the end)
DSNM	DataSet Name	String

Note: for a description of volume names, please refer to the Appendix on "Volume Names" at the end of this document.

3.2 Example FILE:

```
FILE
{
  VLTY ("HFS")
  VLNM ("b4028aef-2")
  DSNM ("14:Track1.trk")
}
```

4 Project Files

A Project file contains info for the project (SampleRate, FrameRate) as well as fields to indicate all the track files that belong to this project, and which track slot it belongs to. The project name attribute (described in next section) must be the first property in a project file.

4.1 Definition of a ProjectFile

```
PJNM '(' "string" ')'  
  
<SimpleProperty>*  
  
TKLS  
{  
  ( TRAK <ComplexValue> ) *  
}  
  
#
```

The following simple properties are used for Projects:

```
SMRT  
FRRT  
SASI  
SNTY  
TKSQ  
PUSQ
```

The following fields are found within the <ComplexValue> of a TRAK:

```
TKNM  
TKIN  
TKOI  
TKDS
```

(see below for explanation of fields)

4.2 ProjectFile FieldNames

```
=====
```

PJNM	Project Name	String
SMRT	Sample Rate	Integer (see below)
FRRT	Frame Rate	Integer (see below)
SASI	Sample Size in Bytes	Integer
SNTY	Sound Type For Record	String ("SDII", "WAV")
PUSQ	Punch Sequence	Integer (last punch sequence used by proj)
TKSQ	Track Sequence	Integer (last track sequence number used)
TKLS	Track List	{ List of TRAK }
TRAK	Track File	{ List of Simple Properties }
TKNM	Track Name	String

TKIN	Track Index	Integer (assigned track output) A value higher than is supported by a system is treated as virtual.
TKOI	Track Original Index	Integer (original assigned track output) To retain the assigned output when a track becomes virtual.
TKDS	Track FILE Reference	FILE complex value

- optional Marker list (MKLS) may follow required Track List (TKLS) -

MKLS	Marker List	{ List of MARK }
MARK	Marker	{ List of Simple Properties }
MKNB	Marker Number	Integer
MKPT	Marker Point	Sample number
MKNM	Marker Name	String

4.3 Values for SMRT (Sample Rate)

```

-----
0 = 44056           // 44100 pulldown
1 = 44100
2 = 44144           // 44100 pullup
3 = 47952           // 48000 pulldown
4 = 48000
5 = 48048           // 48000 pullup
6 = 42294           // 44100 * 23.976 / 25
7 = 42336           // 44100 * 24 / 25
8 = 45938           // 44100 * 25 / 24
9 = 45983           // 44100 * 25 / 23.976
10 = 46034          // 48000 * 23.976 / 25
11 = 46080          // 48000 * 24 / 25
12 = 50000          // 48000 * 25 / 24
13 = 50050          // 48000 * 25 / 23.976

```

For doubled sample rates (88.2K, 96K, etc.) the following multiplier mask is used to indicate that the rate is doubled:

```
SR_MULTIPLIER_2_96 = 0x40
```

So a 96K rate would be represented as 0x44,
88.2K rate would be represented as 0x41.

Values for FRRT (Frame Rates)

```

-----
0 = 30 NDF
1 = 30 DF
2 = 25
3 = 24
4 = 29 NDF
5 = 29 DF

```

Note: Frame Rates are used for display purposes only, for example calculating timecode display. Actual internal calculations of position or lengths must use Sample Rate values.

4.4 Example ProjectFile:

```
-----
PJNM ("MMR Project")
SMRT (48000)
SASI (2)
SNTY ("SDII")
FRRT (0)
PUSQ (2)
TKSQ (2)
TKLS
{
    TRAK
    {
        TKNM ("Track 1")
        TKIN (1)
        TKDS
        {
            FILE
            {
                VLTY ("HFS")
                VLNM ("b4028aef-2")
                DSNM ("14:Track1.trk")
            }
        }
    }
    TRAK
    {
        TKNM ("Track 1")
        TKIN (2)
        TKDS
        {
            FILE
            {
                VLTY ("HFS")
                VLNM ("b4028aef-2")
                DSNM ("14:Track2.trk")
            }
        }
    }
}
MKLS
{
    MARK
    {
        MKNB(1)
        MKPT(172800000)
        MKNM("Marker 1");
    }
    MARK
    {
        MKNB(2)
        MKPT(175680000)
        MKNM("Marker 2");
    }
}
#
```

5 TrackFiles

Track files contain the EDL information for a track as well as track attributes such as track name, track start time. The EDL information is a list of events containing event fields and values.

5.1 Definition of a TrackFile

<SimpleProperty>*

EDLS

```
'{'  
  ( EVNT <ComplexValue> ) *  
}'
```

```
[  
  GNLS  
  {'  
    ( GNEV <ComplexValue> ) *  
  }'  
]
```

```
[  
  MULS  
  {'  
    ( MUEV <ComplexValue> ) *  
  }'  
]  
'#'
```

Note: Items between '[' and ']' are optional

The following simple properties are used for Track Files:

TKNM
STTM

The following fields are found within the <ComplexValue> of a EVNT:

EDNM
EDPT
EDLN
RULN
RUOF
RUSH
RDLN
RDOF
RDSH
INGN
FNGN
EVMU
DSOF
SNTY
EVDS

(see below for explanation of fields)

5.2 *TrackFile FieldNames and descriptions*

TKNM	Track Name	String
STTM	Start Time	Sample Number
EDLS	Edit List	{ List of EVNT }
EVNT	Event	{ List of Simple Properties }
EDNM	Edit Name	String
EDPT	Edit Point	Sample Number
EDLN	Edit Length	Integer (≥ 0)
RULN	RampUp Length	Integer (≥ 0)
RUOF	RampUp Offset	Integer (≥ 0)
RUSH	RampUp Shape	Enumerated Type (0 = Linear)
RDLN	RampDown Length	Integer (≥ 0)
RDOF	RampDown Offset	Integer (≥ 0)
RDSH	RampDown Shape	Enumerated Type (0 = Linear)
INGN	Initial Gain	Floating Point(0.0 - 1.0) Linear Gain (Amplitude)
FNGN	Final Gain	Floating Point(0.0 - 1.0) Linear Gain (Amplitude)
EVMU	Event Muted	0 or 1
DSOF	DatsSet Offset	Sample Number
SNTY	Sound Data Type	String ("SDII", "WAV")
EVDS	Event FILE reference	FILE complex value

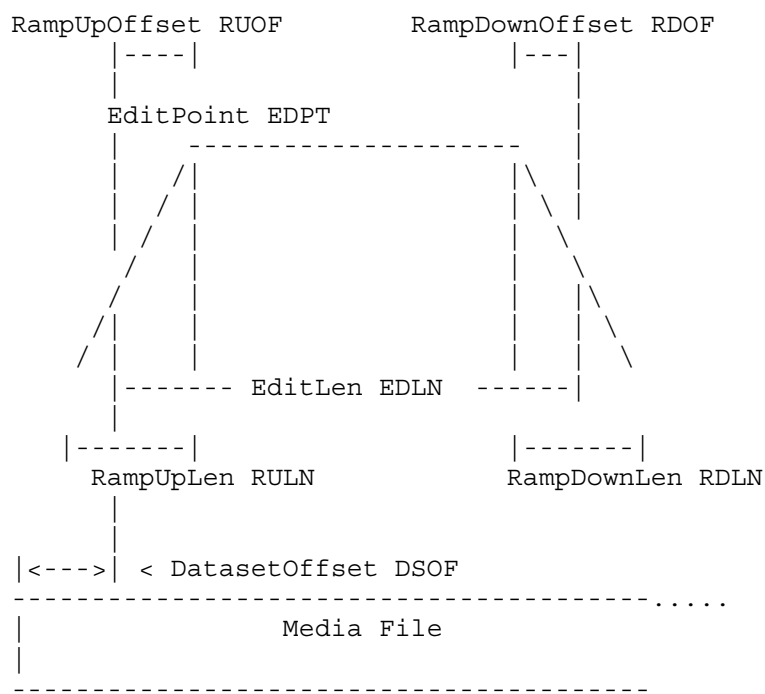
- optional Gain list (GNLS) may follow required EDLS -

GNLS	Gain List (Volume auto.)	{ List of GNEV }
GNEV	Gain Event (volume level)	{ List of Simple Properties }
GNPT	Gain Point	Sample Number
GNLV	Gain Level	Floating Point (0.0 - 1.0) (Slider Value or Fader Level)

- optional Mute list (MULS) may also follow required EDLS -

MULS	Mute List (Mute auto)	{ List of MUEV }
MUEV	Mute Event (Mute On/Off)	{ List of Simple Properties }
MUPT	Mute Point	Sample Number
MULV	Integer	(0 = OFF, 1 = ON)

5.3 Clips and Transitions



RampUpOffset must be \geq zero
RampUpLen must be \geq zero
RampDownOffset must be \geq zero
RampDownLen must be \geq zero

RampUpOffset must be \leq RampUpLen
RampDownOffset must be \leq RampDownLen

First sample to play = DatasetOffset + RampUpOffset - RampUpLen

Length to play = EditLen - RampUpOffset - RampDownOffset + RampUpLen
+ RampDownLen

Clip EditPoints must be in ascending order (No overlapping splice points)

Zero Length clips are NOT allowed

5.4 Slider Value (Fader Level) to Gain Conversion

Slider Value	dB	Amplitude (linear gain)
-----------------	----	----------------------------

1.0	-----	+6 dB2 X gain
	-----	0 dB Unity gain

0.0	-----	-∞ dB0 gain

$$\text{dB} = 40 * \log(10^{6/40 * S}) \quad (\text{dB} = 40 \text{ times log of } 10 \text{ to the } 6/40\text{th times } S)$$

$$\text{Amplitude} = (10^{6/40 * S^2}) \quad (\text{Amp} = 10 \text{ to the } 6/40\text{th times } S \text{ squared})$$

where S is "Slider Value"

5.5 Example TrackFile:

```
TKNM ("Track 1")
STTM (172800000)
EDLS
{
  EVNT
  {
    EDNM ("Cut 1, Track 1")
    EDPT (172910000)
    EDLN (96000)
    RULN (10000)
    RUOF (5000)
    RUSH (0)
    RDLN (10000)
    RDOF (5000)
    RDSH (0)
    INGN (1.000)
    FNGN (1.000)
    EVMU (0)
    DSOF (5000)
    SNTY ("SDII")
    EVDS
    {
      FILE
      {
        VLTY ("HFS")
        VLNM ("b4028aef-2")
        DSNM ("15:Audio 1")
      }
    }
  }
}
```

```

    }
  }
}
EVNT
{
  EDNM ("Cut 2, Track 1")
  EDPT (173910000)
  EDLN (48000)
  RULN (480)
  RUOF (240)
  RUSH (0)
  RDLN (480)
  RDOF (240)
  RDSH (0)
  INGN (1.000)
  FNGN (1.000)
  EVMU (0)
  DSOF (240)
  SNTY ("SDII")
  EVDS
  {
    FILE
    {
      VLTY ("HFS")
      VLNM ("b4028aef-2")
      DSNM ("15:Audio 2")
    }
  }
}
}
GNLS
{
  GNEV
  {
    GNPT (172800000)
    GNLV (1.00)
  }
  GNEV
  {
    GNPT (172900000)
    GNLV (0.50)
  }
}
MULS
{
  MUEV
  {
    MUPT (173000000)
    MULV (1)
  }
  MUEV
  {
    MUPT (173500000)
    MULV (0)
  }
}
#

```

6 APPENDIX

=====

6.1 Volume Names

6.1.1 HFS Standard/HFS Plus (Extended File Format)

The volume name for Macintosh volumes is comprised of a volume creation date and the partition number for that partition. Partition numbers run from 1..n where n is the number of the last partition on the disk. Creation dates on an HFS volume are unsigned 32 bit integers reflecting standard local time on the system on which the disks were formatted. The volume creation date resides at an offset of 2 from the start of the volume information block laid down by the Mac File System when the disk is partitioned. There is one volume information block for each partition on the disk. The Mac OS currently embeds an HFS Plus volume inside of a standard Mac OS file system when partitioning a Mac OS Extended File System for compatibility reasons. The creation date used for the volume name of an HFS Plus disk is the creation date in the volume information block of the HFS Plus file system, not the creation date of the Standard HFS wrapper. In order to find the volume information block of an HFS Plus volume, you must first read the volume information block for the wrapper which is used to determine the offset to the real start of the embedded HFS Plus volume. Examples of a volume name for a disk in either format would be "b4028ae9-1" for partition 1, "b4028aef-2" for partition 2, etc.

6.1.2 FAT32

The volume name for a FAT32 disk is comprised of a string representing the hexadecimal value of the volume serial number for that disk. The volume serial number is a unsigned 32 bit field laid down by the format routine and stored at offset 0x027 from the beginning of the disk (sector 0). The volume serial number will change or be updated any time the volume is reformatted. An example of the volume name for a FAT16 or FAT32 disk would be "12F637B8".

6.2 Directory Names

By convention, the following directory (folder) names will be used for OpenTL projects:

```
<MyProject>                // Any reasonable project name
    MyProject.TL
    <Track Files>            // By convention "Track Files"
        MyTrack1             // Track files....
        MyTrack2
    <Audio Files>            // By convention "Audio Files"
        Audio1               // Audio files....
        Audio2
```