cellar
Internet-Draft
Intended status: Informational
Expires: 22 April 2021

S. Lhomme

M. Bunkus

D. Rice

19 October 2020

Matroska Media Container Format Specifications
draft-ietf-cellar-matroska-06

Abstract

This document defines the Matroska audiovisual container, including definitions of its structural elements, as well as its terminology, vocabulary, and application.

Table of Contents

1. Introduction

Matroska aims to become THE standard of multimedia container formats. It was derived from a project called [MCF], but differentiates from it significantly because it is based on EBML (Extensible Binary Meta Language) [RFC8794], a binary derivative of XML. EBML enables significant advantages in terms of future format extensibility, without breaking file support in old parsers.

First, it is essential to clarify exactly "What an Audio/Video container is", to avoid any misunderstandings:

* It is NOT a video or audio compression format (codec)

* It is an envelope for which there can be many audio, video, and subtitles streams, allowing the user to store a complete movie or CD in a single file.

Matroska is designed with the future in mind. It incorporates features like:

* Fast seeking in the file
* Chapter entries
* Full metadata (tags) support
* Selectable subtitle/audio/video streams
* Modularly expandable
* Error resilience (can recover playback even when the stream is damaged)
* Streamable over the internet and local networks (HTTP, CIFS, FTP, etc)
* Menus (like DVDs have)

Matroska is an open standards project. This means for personal use it is absolutely free to use and that the technical specifications describing the bitstream are open to everybody, even to companies that would like to support it in their products.

2. Status of this document

This document is a work-in-progress specification defining the Matroska file format as part of the IETF Cellar working group (https://datatracker.ietf.org/wg/cellar/charter/). But since it's quite complete it is used as a reference for the development of libmatroska. A legacy version of the specification can be found here (https://www.matroska.org/files/matroska_file_format_alexander_noe.pdf) (PDF doc by Alexander Noé -- outdated).

For a simplified diagram of the layout of a Matroska file, see the Diagram page (diagram.md).

The table found below is now generated from the "source" of the Matroska specification. This XML file (https://github.com/Matroska-Org/foundation-source/blob/master/spectool/specdata.xml) is also used to generate the semantic data used in libmatroska and libmatroska2. We encourage anyone to use and monitor its changes so your code is spec-proof and always up to date.

Note that versions 1, 2, and 3 have been finalized. Version 4 is currently work in progress. There MAY be further additions to v4.

3. Security Considerations

Matroska inherits security considerations from EBML.

Attacks on a "Matroska Reader" could include:

* Storage of a arbitrary and potentially executable data within an "Attachment Element". "Matroska Readers" that extract or use data from Matroska Attachments SHOULD check that the data adheres to expectations.

* A "Matroska Attachment" with an inaccurate mime-type.

4. IANA Considerations

To be determined.

5. Notation and Conventions

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "NOT RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in BCP 14 [RFC2119] [RFC8174] when, and only when, they appear in all capitals, as shown here.

This document defines specific terms in order to define the format and application of "Matroska". Specific terms are defined below:

"Matroska": A multimedia container format based on EBML (Extensible Binary Meta Language).

"Matroska Reader": A data parser that interprets the semantics of a Matroska document and creates a way for programs to use "Matroska".

"Matroska Player": A "Matroska Reader" with a primary purpose of playing audiovisual files, including "Matroska" documents.

6. Basis in EBML

Matroska is a Document Type of EBML (Extensible Binary Meta Language). This specification is dependent on the EBML Specification [RFC8794]. For an understanding of Matroska's EBML Schema, see in particular the sections of the EBML Specification covering EBML Element Types (Section 7), EBML Schema (Section 11.1), and EBML Structure (Section 3).

6.1. Added Constraints on EBML

As an EBML Document Type, Matroska adds the following constraints to the EBML specification.

* The "docType" of the "EBML Header" MUST be "matroska".

* The "EBMLMaxIDLength" of the "EBML Header" MUST be "4".

* The "EBMLMaxSizeLength" of the "EBML Header" MUST be between "1" and "8" inclusive.

6.2. Matroska Design

All top-levels elements (Segment and direct sub-elements) are coded on 4 octets -- i.e. class D elements.

6.2.1. Language Codes

Matroska from version 1 through 3 uses language codes that can be either the 3 letters bibliographic ISO-639-2 (https://www.loc.gov/standards/iso639-2/php/English_list.php) form (like "fre" for french), or such a language code followed by a dash and a country code for specialities in languages (like "fre-ca" for Canadian French). The "ISO 639-2 Language Elements" are "Language Element", "TagLanguage Element", and "ChapLanguage Element".

Starting in Matroska version 4, either "ISO 639-2" or BCP 47 (https://tools.ietf.org/html/bcp47) MAY be used, although "BCP 47" is RECOMMENDED. The "BCP 47 Language Elements" are "LanguageIETF Element", "TagLanguageIETF Element", and "ChapLanguageIETF Element". If a "BCP 47 Language Element" and an "ISO 639-2 Language Element" are used within the same "Parent Element", then the "ISO 639-2 Language Element" MUST be ignored and precedence given to the "BCP 47 Language Element".

Country codes are the same as used for internet domains (https://www.iana.org/domains/root/db).

6.2.2. Physical Types

Each level can have different meanings for audio and video. The ORIGINAL_MEDIUM tag can be used to specify a string for ChapterPhysicalEquiv = 60. Here is the list of possible levels for both audio and video:

ChapterPhysicalEquiv	Audio	Video	Comment
70	SET / PACKAGE	SET / PACKAGE	the collection of different media
60	CD / 12" / 10" / 7" / TAPE / MINIDISC / DAT	DVD / VHS / LASERDISC	the physical medium like a CD or a DVD
50	SIDE	SIDE	when the original medium (LP/DVD) has different sides
40	-	LAYER	another physical level on DVDs
30	SESSION	SESSION	as found on CDs and DVDs
20	TRACK	-	as found on audio CDs
10	INDEX	-	the first logical level of the side/ medium

Table 1

6.2.3. Block Structure

Bit 0 is the most significant bit.

Frames using references SHOULD be stored in "coding order". That means the references first, and then the frames referencing them. A consequence is that timestamps might not be consecutive. But a frame with a past timestamp MUST reference a frame already known, otherwise it's considered bad/void.

6.2.3.1. Block Header

Offset	Player	Description
0x00+	MUST	Track Number (Track Entry). It is coded in EBML like form (1 octet if the value is < 0x80, 2 if < 0x4000, etc) (most significant bits set to increase the range).
0x01+	MUST	Timestamp (relative to Cluster timestamp, signed int16)

Table 2

6.2.3.2. Block Header Flags

Offset	Bit	Player	Description
0x03+	0-3	-	Reserved, set to 0
0x03+	4	-	Invisible, the codec SHOULD decode this frame but not display it
0x03+	5-6	MUST	Lacing
			* 00 : no lacing
			* 01 : Xiph lacing
			* 11 : EBML lacing
			* 10 : fixed-size lacing
0x03+	7	-	not used

Table 3

6.2.4. Lacing

Lacing is a mechanism to save space when storing data. It is typically used for small blocks of data (referred to as frames in Matroska). There are 3 types of lacing:

1. Xiph, inspired by what is found in the Ogg container
2. EBML, which is the same with sizes coded differently
3. fixed-size, where the size is not coded

For example, a user wants to store 3 frames of the same track. The first frame is 800 octets long, the second is 500 octets long and the third is 1000 octets long. As these data are small, they can be stored in a lace to save space. They will then be stored in the same block as follows:

6.2.4.1. Xiph lacing

* Block head (with lacing bits set to 01)
* Lacing head: Number of frames in the lace -1 -- i.e. 2 (the 800 and 500 octets one)
* Lacing sizes: only the 2 first ones will be coded, 800 gives 255;255;255;35, 500 gives 255;245. The size of the last frame is deduced from the total size of the Block.
* Data in frame 1
* Data in frame 2
* Data in frame 3

A frame with a size multiple of 255 is coded with a 0 at the end of the size -- for example, 765 is coded 255;255;255;0.

6.2.4.2. EBML lacing

In this case, the size is not coded as blocks of 255 bytes, but as a difference with the previous size and this size is coded as in EBML. The first size in the lace is unsigned as in EBML. The others use a range shifting to get a sign on each value:

Bit Representation	Value
1xxx xxxx	value -(2^(6)-1) to 2^(6)-1 (ie 0 to 2^(7)-2 minus 2^(6)-1, half of the range)
01xx xxxx xxxx xxxx	value -(2^(13)-1) to 2^(13)-1
001x xxxx xxxx xxxx xxxx xxxx	value -(2^(20)-1) to 2^(20)-1
0001 xxxx xxxx xxxx xxxx xxxx xxxx xxxx	value -(2^(27)-1) to 2^(27)-1
0000 1xxx xxxx xxxx xxxx xxxx xxxx xxxx xxxx xxxx	value -(2^(34)-1) to 2^(34)-1
0000 01xx xxxx xxxx xxxx xxxx xxxx xxxx xxxx xxxx xxxx xxxx	value -(2^(41)-1) to 2^(41)-1
0000 001x xxxx xxxx xxxx xxxx xxxx xxxx xxxx xxxx xxxx xxxx xxxx	value -(2^(48)-1) to 2^(48)-1

Table 4

* Block head (with lacing bits set to 11)

* Lacing head: Number of frames in the lace -1 -- i.e. 2 (the 800 and 500 octets one)

* Lacing sizes: only the 2 first ones will be coded, 800 gives 0x320 0x4000 = 0x4320, 500 is coded as -300 : - 0x12C + 0x1FFF + 0x4000 = 0x5ED3. The size of the last frame is deduced from the total size of the Block.

* Data in frame 1

* Data in frame 2

* Data in frame 3

6.2.4.3. Fixed-size lacing

In this case, only the number of frames in the lace is saved, the size of each frame is deduced from the total size of the Block. For example, for 3 frames of 800 octets each:

* Block head (with lacing bits set to 10)
* Lacing head: Number of frames in the lace -1 -- i.e. 2
* Data in frame 1
* Data in frame 2
* Data in frame 3

6.2.4.4. SimpleBlock Structure

The "SimpleBlock" is inspired by the Block structure; see Section 6.2.3. The main differences are the added Keyframe flag and Discardable flag. Otherwise everything is the same.

Bit 0 is the most significant bit.

Frames using references SHOULD be stored in "coding order". That means the references first, and then the frames referencing them. A consequence is that timestamps might not be consecutive. But a frame with a past timestamp MUST reference a frame already known, otherwise it's considered bad/void.

6.2.4.4.1. SimpleBlock Header

Offset	Player	Description
0x00+	MUST	Track Number (Track Entry). It is coded in EBML like form (1 octet if the value is < 0x80, 2 if < 0x4000, etc) (most significant bits set to increase the range).
0x01+	MUST	Timestamp (relative to Cluster timestamp, signed int16)

Table 5

6.2.4.4.2. SimpleBlock Header Flags

Offset	Bit	Player	Description
0x03+	0	-	Keyframe, set when the Block contains only keyframes
0x03+	1-3	-	Reserved, set to 0
0x03+	4	-	Invisible, the codec SHOULD decode this frame but not display it
0x03+	5-6	MUST	Lacing
			* 00 : no lacing
			* 01 : Xiph lacing
			* 11 : EBML lacing
			* 10 : fixed-size lacing
0x03+	7	-	Discardable, the frames of the Block can be discarded during playing if needed

Table 6

6.2.4.5. Laced Data

When lacing bit is set.

Offset	Player	Description
0x00	MUST	Number of frames in the lace-1 (uint8)
0x01 / 0xXX	MUST*	Lace-coded size of each frame of the lace, except for the last one (multiple uint8). *This is not used with Fixed-size lacing as it is calculated automatically from (total size of lace) / (number of frames in lace).

Table 7

For (possibly) Laced Data

Offset	Player	Description
0x00	MUST	Consecutive laced frames

Table 8

7. Matroska Structure

A Matroska file MUST be composed of at least one "EBML Document" using the "Matroska Document Type". Each "EBML Document" MUST start with an "EBML Header" and MUST be followed by the "EBML Root Element", defined as "Segment" in Matroska. Matroska defines several "Top Level Elements" which MAY occur within the "Segment".

As an example, a simple Matroska file consisting of a single "EBML Document" could be represented like this:

* "EBML Header"

* "Segment"

A more complex Matroska file consisting of an "EBML Stream" (consisting of two "EBML Documents") could be represented like this:

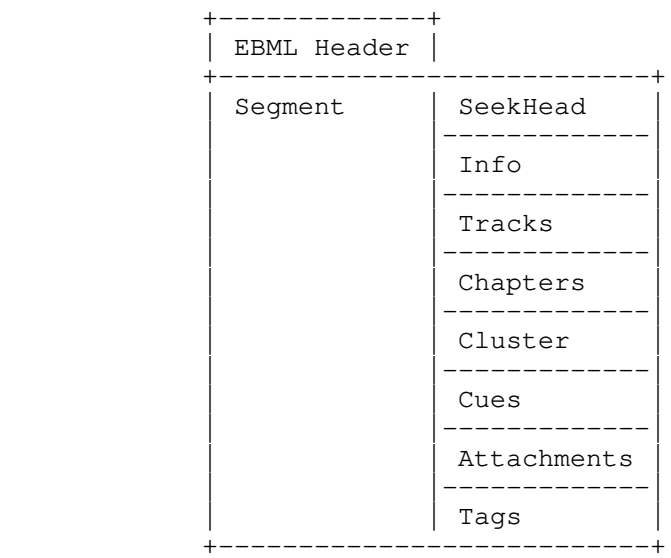
* "EBML Header"

* "Segment"

* "EBML Header"

* "Segment"

The following diagram represents a simple Matroska file, comprised of an "EBML Document" with an "EBML Header", a "Segment Element" (the "Root Element"), and all eight Matroska "Top Level Elements". In the following diagrams of this section, horizontal spacing expresses a parent-child relationship between Matroska Elements (e.g., the "Info Element" is contained within the "Segment Element") whereas vertical alignment represents the storage order within the file.



The Matroska "EBML Schema" defines eight "Top Level Elements": "SeekHead", "Info", "Tracks", "Chapters", "Cluster", "Cues", "Attachments", and "Tags".

The "SeekHead Element" (also known as "MetaSeek") contains an index of "Top Level Elements" locations within the "Segment". Use of the "SeekHead Element" is RECOMMENDED. Without a "SeekHead Element", a Matroska parser would have to search the entire file to find all of the other "Top Level Elements". This is due to Matroska's flexible ordering requirements; for instance, it is acceptable for the "Chapters Element" to be stored after the "Cluster Elements".

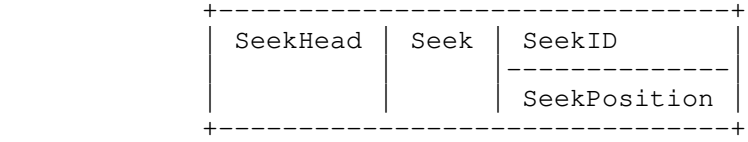


Figure 1: Representation of a "SeekHead Element".

The "Info Element" contains vital information for identifying the whole "Segment". This includes the title for the "Segment", a randomly generated unique identifier, and the unique identifier(s) of any linked "Segment Elements".

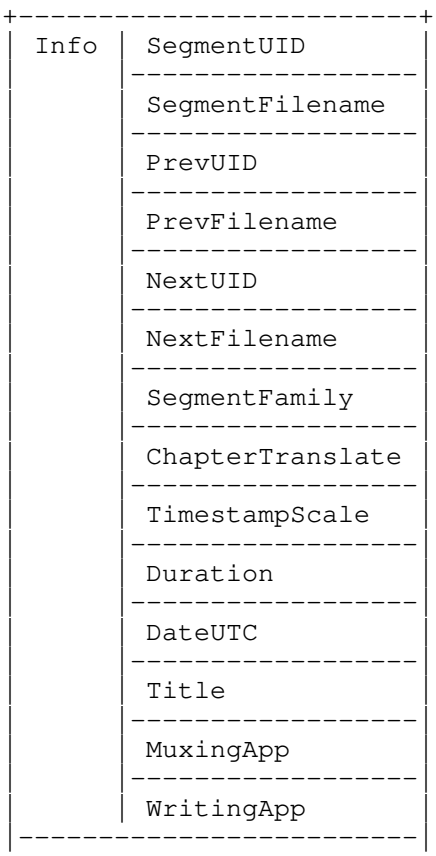


Figure 2: Representation of an "Info Element" and its "Child Elements".

The "Tracks Element" defines the technical details for each track and can store the name, number, unique identifier, language, and type (audio, video, subtitles, etc.) of each track. For example, the "Tracks Element" MAY store information about the resolution of a video track or sample rate of an audio track.

The "Tracks Element" MUST identify all the data needed by the codec to decode the data of the specified track. However, the data required is contingent on the codec used for the track. For example, a "Track Element" for uncompressed audio only requires the audio bit rate to be present. A codec such as AC-3 would require that the "CodecID Element" be present for all tracks, as it is the primary way to identify which codec to use to decode the track.

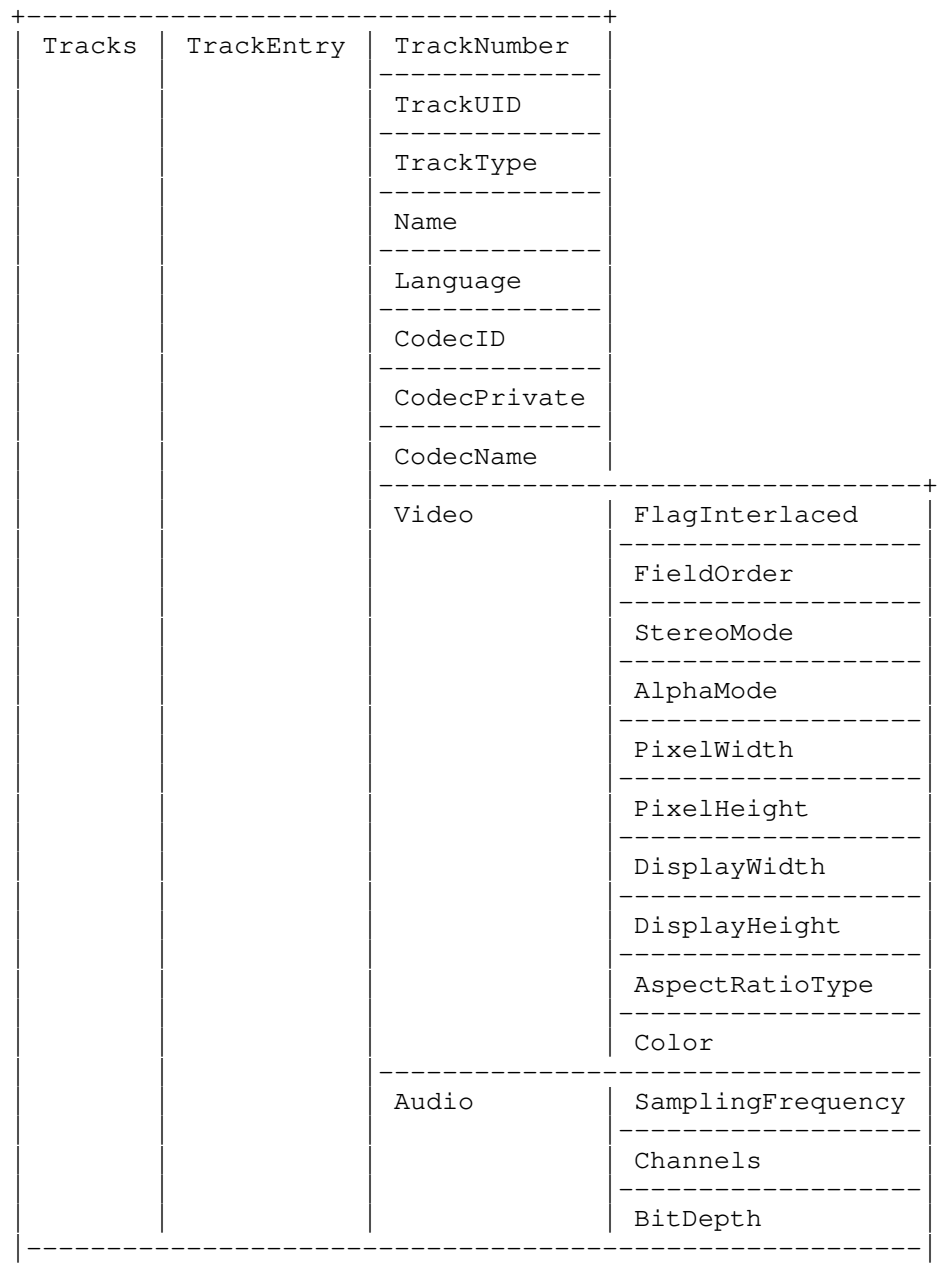


Figure 3: Representation of the "Tracks Element" and a selection of its "Descendant Elements".

The "Chapters Element" lists all of the chapters. Chapters are a way to set predefined points to jump to in video or audio.

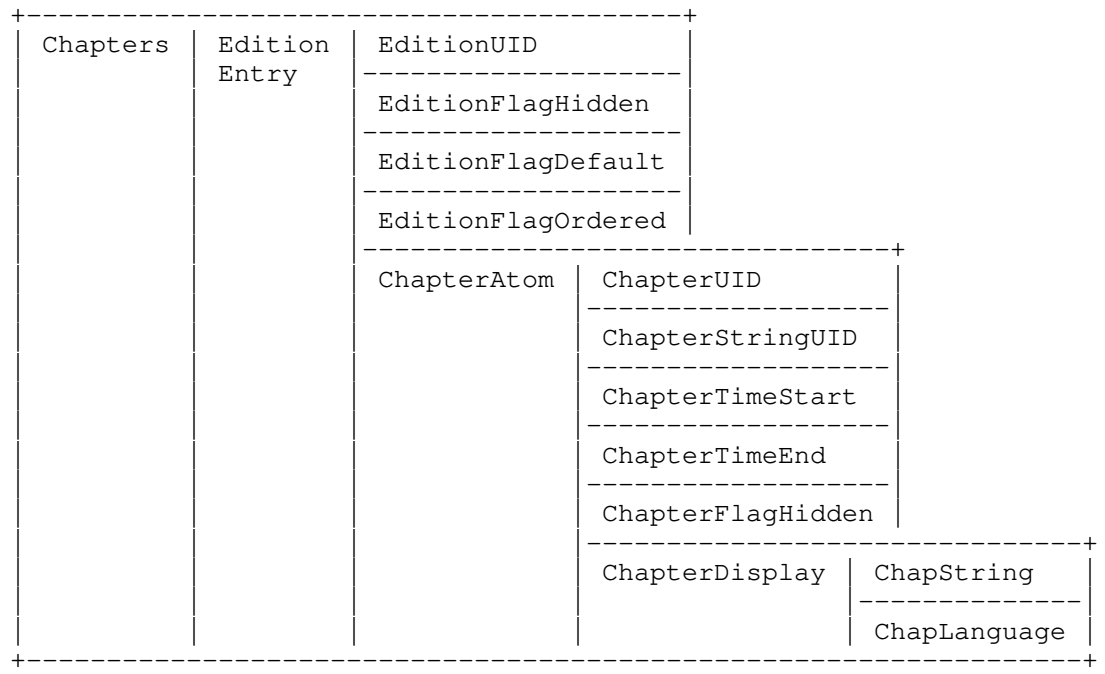


Figure 4: Representation of the "Chapters Element" and a selection of its "Descendant Elements".

"Cluster Elements" contain the content for each track, e.g., video frames. A Matroska file SHOULD contain at least one "Cluster Element". The "Cluster Element" helps to break up "SimpleBlock" or "BlockGroup Elements" and helps with seeking and error protection. It is RECOMMENDED that the size of each individual "Cluster Element" be limited to store no more than 5 seconds or 5 megabytes. Every "Cluster Element" MUST contain a "Timestamp Element". This SHOULD be the "Timestamp Element" used to play the first "Block" in the "Cluster Element". There SHOULD be one or more "BlockGroup" or "SimpleBlock Element" in each "Cluster Element". A "BlockGroup Element" MAY contain a "Block" of data and any information relating directly to that "Block".

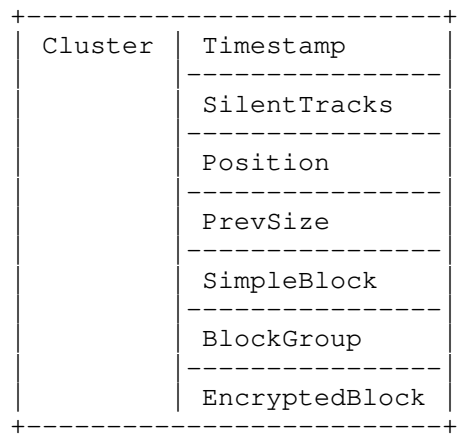


Figure 5: Representation of a "Cluster Element" and its immediate "Child Elements".

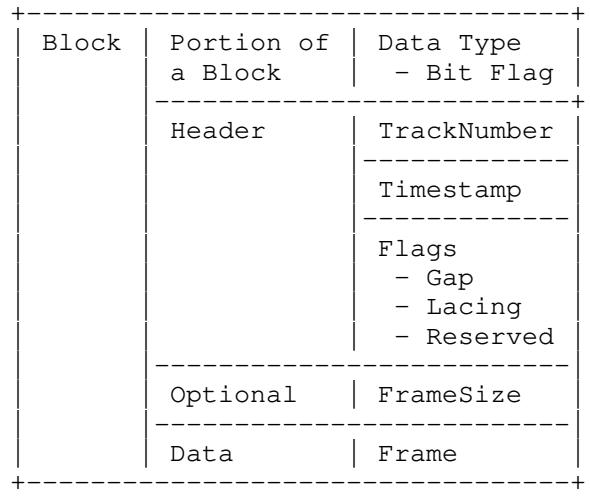


Figure 6: Representation of the "Block Element" structure.

Each "Cluster" MUST contain exactly one "Timestamp Element". The "Timestamp Element" value MUST be stored once per "Cluster". The "Timestamp Element" in the "Cluster" is relative to the entire "Segment". The "Timestamp Element" SHOULD be the first "Element" in the "Cluster".

Additionally, the "Block" contains an offset that, when added to the "Cluster"'s "Timestamp Element" value, yields the "Block"'s effective timestamp. Therefore, timestamp in the "Block" itself is relative to

the "Timestamp Element" in the "Cluster". For example, if the "Timestamp Element" in the "Cluster" is set to 10 seconds and a "Block" in that "Cluster" is supposed to be played 12 seconds into the clip, the timestamp in the "Block" would be set to 2 seconds.

The "ReferenceBlock" in the "BlockGroup" is used instead of the basic "P-frame"/"B-frame" description. Instead of simply saying that this "Block" depends on the "Block" directly before, or directly afterwards, the "Timestamp" of the necessary "Block" is used. Because there can be as many "ReferenceBlock Elements" as necessary for a "Block", it allows for some extremely complex referencing.

The "Cues Element" is used to seek when playing back a file by providing a temporal index for some of the "Tracks". It is similar to the "SeekHead Element", but used for seeking to a specific time when playing back the file. It is possible to seek without this element, but it is much more difficult because a "Matroska Reader" would have to 'hunt and peck' through the file looking for the correct timestamp.

The "Cues Element" SHOULD contain at least one "CuePoint Element". Each "CuePoint Element" stores the position of the "Cluster" that contains the "BlockGroup" or "SimpleBlock Element". The timestamp is stored in the "CueTime Element" and location is stored in the "CueTrackPositions Element".

The "Cues Element" is flexible. For instance, "Cues Element" can be used to index every single timestamp of every "Block" or they can be indexed selectively. For video files, it is RECOMMENDED to index at least the keyframes of the video track.

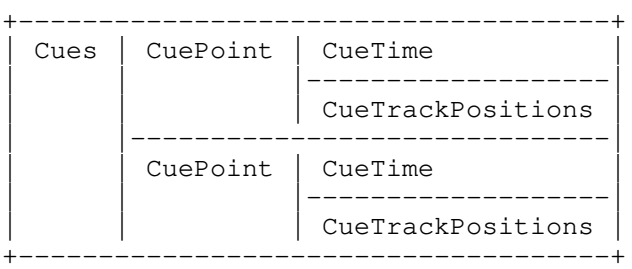


Figure 7: Representation of a "Cues Element" and two levels of its "Descendant Elements".

The "Attachments Element" is for attaching files to a Matroska file such as pictures, webpages, programs, or even the codec needed to play back the file.

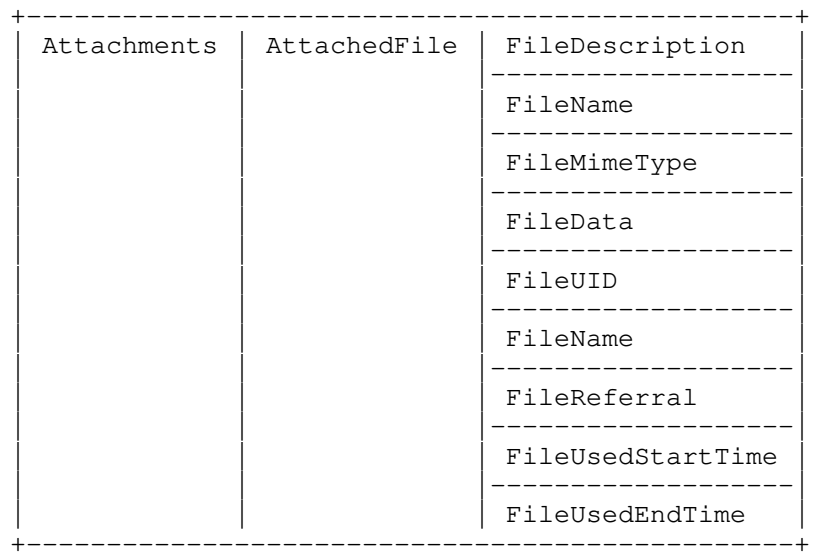


Figure 8: Representation of a "Attachments Element".

The "Tags Element" contains metadata that describes the "Segment" and potentially its "Tracks", "Chapters", and "Attachments". Each "Track" or "Chapter" that those tags applies to has its UID listed in the "Tags". The "Tags" contain all extra information about the file: scriptwriter, singer, actors, directors, titles, edition, price, dates, genre, comments, etc. Tags can contain their values in multiple languages. For example, a movie's "title" "Tag" might contain both the original English title as well as the title it was released as in Germany.

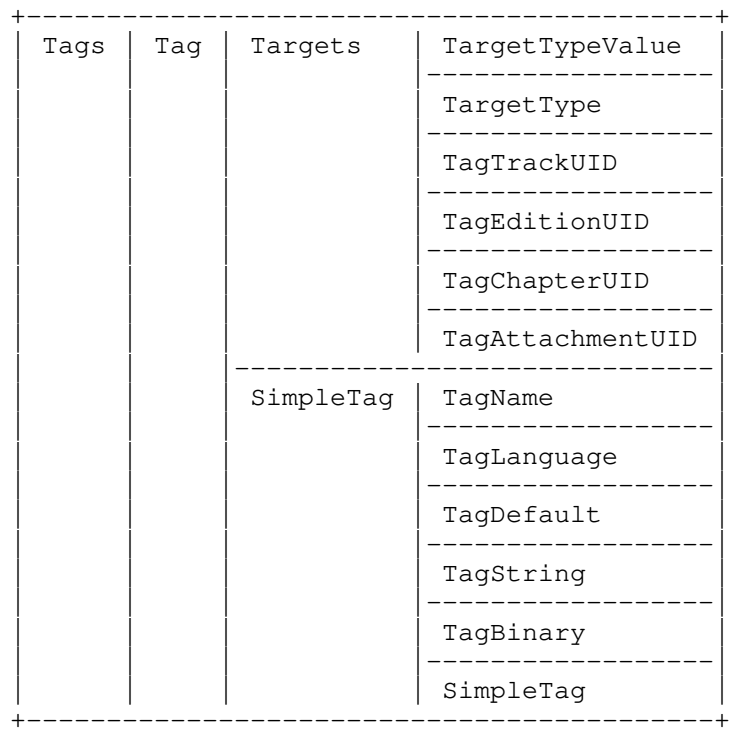


Figure 9: Representation of a "Tags Element" and three levels of its "Children Elements".

8. Matroska Additions to Schema Element Attributes

In addition to the EBML Schema definition provided by the EBML Specification, Matroska adds the following additional attributes:

attribute name	required	definition
webm	No	A boolean to express if the Matroska Element is also supported within version 2 of the "webm" specification. Please consider the webm specification (http://www.webmproject.org/docs/container/) as the authoritative on "webm".

Table 9

9. Matroska Schema

This specification includes an "EBML Schema", which defines the Elements and structure of Matroska as an EBML Document Type. The EBML Schema defines every valid Matroska element in a manner defined by the EBML specification.

Here the definition of each Matroska Element is provided.

9.1. EBMLMaxIDLength Element

name: EBMLMaxIDLength

path: "\EBML\EBMLMaxIDLength"

id: 0x42F2

minOccurs: 1

maxOccurs: 1

range: 4

default: 4

type: uinteger

9.2. EBMLMaxSizeLength Element

name: EBMLMaxSizeLength

path: "\EBML\EBMLMaxSizeLength"

id: 0x42F3

minOccurs: 1

maxOccurs: 1

range: 1-8

default: 8

type: uinteger

10. Segment Element

name: Segment

path: "\Segment"

id: 0x18538067

minOccurs: 1

maxOccurs: 1

type: master

unknownsizeallowed: 1

definition: The Root Element that contains all other Top-Level Elements (Elements defined only at Level 1). A Matroska file is composed of 1 Segment.

10.1. SeekHead Element

name: SeekHead

path: "\Segment\SeekHead"

id: 0x114D9B74

maxOccurs: 2

type: master

definition: Contains the Segment Position of other Top-Level Elements.

10.1.1. Seek Element

name: Seek

path: "\Segment\SeekHead\Seek"

id: 0x4DBB

minOccurs: 1

type: master

definition: Contains a single seek entry to an EBML Element.

10.1.1.1. SeekID Element

name: SeekID

path: "\Segment\SeekHead\Seek\SeekID"

id: 0x53AB

minOccurs: 1

maxOccurs: 1

type: binary

definition: The binary ID corresponding to the Element name.

10.1.1.2. SeekPosition Element

name: SeekPosition

path: "\Segment\SeekHead\Seek\SeekPosition"

id: 0x53AC

minOccurs: 1

maxOccurs: 1

type: uinteger

definition: The Segment Position of the Element.

10.2. Info Element

name: Info

path: "\Segment\Info"

id: 0x1549A966

minOccurs: 1

type: master

recurring: 1

definition: Contains general information about the Segment.

10.2.1. SegmentUID Element

name: SegmentUID

path: "\Segment\Info\SegmentUID"

id: 0x73A4

maxOccurs: 1

range: not 0

type: binary

definition: A randomly generated unique ID to identify the Segment amongst many others (128 bits).

usage notes: If the Segment is a part of a Linked Segment, then this Element is REQUIRED.

10.2.2. SegmentFilename Element

name: SegmentFilename

path: "\Segment\Info\SegmentFilename"

id: 0x7384

maxOccurs: 1

type: utf-8

definition: A filename corresponding to this Segment.

10.2.3. PrevUID Element

name: PrevUID

path: "\Segment\Info\PrevUID"

id: 0x3CB923

maxOccurs: 1

type: binary

definition: A unique ID to identify the previous Segment of a Linked Segment (128 bits).

usage notes: If the Segment is a part of a Linked Segment that uses Hard Linking, then either the PrevUID or the NextUID Element is REQUIRED. If a Segment contains a PrevUID but not a NextUID, then it MAY be considered as the last Segment of the Linked Segment. The PrevUID MUST NOT be equal to the SegmentUID.

10.2.4. PrevFilename Element

name: PrevFilename

path: "\Segment\Info\PrevFilename"

id: 0x3C83AB

maxOccurs: 1

type: utf-8

definition: A filename corresponding to the file of the previous Linked Segment.

usage notes: Provision of the previous filename is for display convenience, but PrevUID SHOULD be considered authoritative for identifying the previous Segment in a Linked Segment.

10.2.5. NextUID Element

name: NextUID

path: "\Segment\Info\NextUID"

id: 0x3EB923

maxOccurs: 1

type: binary

definition: A unique ID to identify the next Segment of a Linked Segment (128 bits).

usage notes: If the Segment is a part of a Linked Segment that uses Hard Linking, then either the PrevUID or the NextUID Element is REQUIRED. If a Segment contains a NextUID but not a PrevUID, then it MAY be considered as the first Segment of the Linked Segment. The NextUID MUST NOT be equal to the SegmentUID.

10.2.6. NextFilename Element

name: NextFilename

path: "\Segment\Info\NextFilename"

id: 0x3E83BB

maxOccurs: 1

type: utf-8

definition: A filename corresponding to the file of the next Linked Segment.

usage notes: Provision of the next filename is for display convenience, but NextUID SHOULD be considered authoritative for identifying the Next Segment.

10.2.7. SegmentFamily Element

name: SegmentFamily

path: "\Segment\Info\SegmentFamily"

id: 0x4444

type: binary

definition: A randomly generated unique ID that all Segments of a Linked Segment MUST share (128 bits).

usage notes: If the Segment is a part of a Linked Segment that uses Soft Linking, then this Element is REQUIRED.

10.2.8. ChapterTranslate Element

name: ChapterTranslate

path: "\Segment\Info\ChapterTranslate"

id: 0x6924

type: master

definition: A tuple of corresponding ID used by chapter codecs to represent this Segment.

10.2.8.1. ChapterTranslateEditionUID Element

name: ChapterTranslateEditionUID

path: "\Segment\Info\ChapterTranslate\ChapterTranslateEditionUID"

id: 0x69FC

type: uinteger

definition: Specify an edition UID on which this correspondence applies. When not specified, it means for all editions found in the Segment.

10.2.8.2. ChapterTranslateCodec Element

name: ChapterTranslateCodec

path: "\Segment\Info\ChapterTranslate\ChapterTranslateCodec"

id: 0x69BF

minOccurs: 1

maxOccurs: 1

type: uinteger

definition: The chapter codec; see Section 10.7.1.5.12.1.

restrictions:

value	label
0	Matroska Script
1	DVD-menu

Table 10

10.2.8.3. ChapterTranslateID Element

name: ChapterTranslateID

path: "\Segment\Info\ChapterTranslate\ChapterTranslateID"

id: 0x69A5

minOccurs: 1

maxOccurs: 1

type: binary

definition: The binary value used to represent this Segment in the chapter codec data. The format depends on the ChapProcessCodecID used; see Section 10.7.1.5.12.1.

10.2.9. TimestampScale Element

name: TimestampScale

path: "\Segment\Info\TimestampScale"

id: 0x2AD7B1

minOccurs: 1

maxOccurs: 1

range: not 0

default: 1000000

type: uinteger

definition: Timestamp scale in nanoseconds (1.000.000 means all timestamps in the Segment are expressed in milliseconds).

10.2.10. Duration Element

name: Duration

path: "\Segment\Info\Duration"

id: 0x4489

maxOccurs: 1

range: > 0x0p+0

type: float

definition: Duration of the Segment in nanoseconds based on

TimestampScale.

10.2.11. DateUTC Element

name: DateUTC

path: "\Segment\Info\DateUTC"

id: 0x4461

maxOccurs: 1

type: date

definition: The date and time that the Segment was created by the muxing application or library.

10.2.12. Title Element

name: Title

path: "\Segment\Info\Title"

id: 0x7BA9

maxOccurs: 1

type: utf-8

definition: General name of the Segment.

10.2.13. MuxingApp Element

name: MuxingApp

path: "\Segment\Info\MuxingApp"

id: 0x4D80

minOccurs: 1

maxOccurs: 1

type: utf-8

definition: Muxing application or library (example: "libmatroska-0.4.3").

usage notes: Include the full name of the application or library followed by the version number.

10.2.14. WritingApp Element

name: WritingApp

path: "\Segment\Info\WritingApp"

id: 0x5741

minOccurs: 1

maxOccurs: 1

type: utf-8

definition: Writing application (example: "mkvmerge-0.3.3").

usage notes: Include the full name of the application followed by the version number.

10.3. Cluster Element

name: Cluster

path: "\Segment\Cluster"

id: 0x1F43B675

type: master

unknownsizeallowed: 1

definition: The Top-Level Element containing the (monolithic) Block structure.

10.3.1. Timestamp Element

name: Timestamp

path: "\Segment\Cluster\Timestamp"

id: 0xE7

minOccurs: 1

maxOccurs: 1

type: uinteger

definition: Absolute timestamp of the cluster (based on TimestampScale).

10.3.2. SilentTracks Element

name: SilentTracks

path: "\Segment\Cluster\SilentTracks"

id: 0x5854

maxOccurs: 1

type: master

definition: The list of tracks that are not used in that part of the stream. It is useful when using overlay tracks on seeking or to decide what track to use.

10.3.2.1. SilentTrackNumber Element

name: SilentTrackNumber

path: "\Segment\Cluster\SilentTracks\SilentTrackNumber"

id: 0x58D7

type: uinteger

definition: One of the track number that are not used from now on in the stream. It could change later if not specified as silent in a further Cluster.

10.3.3. Position Element

name: Position

path: "\Segment\Cluster\Position"

id: 0xA7

maxOccurs: 1

type: uinteger

definition: The Segment Position of the Cluster in the Segment (0 in

live streams). It might help to resynchronise offset on damaged streams.

10.3.4. PrevSize Element

name: PrevSize

path: "\Segment\Cluster\PrevSize"

id: 0xAB

maxOccurs: 1

type: uinteger

definition: Size of the previous Cluster, in octets. Can be useful for backward playing.

10.3.5. SimpleBlock Element

name: SimpleBlock

path: "\Segment\Cluster\SimpleBlock"

id: 0xA3

type: binary

minver: 2

definition: Similar to Block, see Section 6.2.3, but without all the extra information, mostly used to reduced overhead when no extra feature is needed; see Section 6.2.4.4 on SimpleBlock Structure.

10.3.6. BlockGroup Element

name: BlockGroup

path: "\Segment\Cluster\BlockGroup"

id: 0xA0

type: master

definition: Basic container of information containing a single Block and information specific to that Block.

10.3.6.1. Block Element

name: Block

path: "\Segment\Cluster\BlockGroup\Block"

id: 0xA1

minOccurs: 1

maxOccurs: 1

type: binary

definition: Block containing the actual data to be rendered and a timestamp relative to the Cluster Timestamp; see Section 6.2.3 on Block Structure.

10.3.6.2. BlockVirtual Element

name: BlockVirtual

path: "\Segment\Cluster\BlockGroup\BlockVirtual"

id: 0xA2

maxOccurs: 1

type: binary

minver: 0

maxver: 0

definition: A Block with no data. It MUST be stored in the stream at the place the real Block would be in display order.

10.3.6.3. BlockAdditions Element

name: BlockAdditions

path: "\Segment\Cluster\BlockGroup\BlockAdditions"

id: 0x75A1

maxOccurs: 1

type: master

definition: Contain additional blocks to complete the main one. An EBML parser that has no knowledge of the Block structure could still see and use/skip these data.

10.3.6.3.1. BlockMore Element

name: BlockMore

path: "\Segment\Cluster\BlockGroup\BlockAdditions\BlockMore"

id: 0xA6

minOccurs: 1

type: master

definition: Contain the BlockAdditional and some parameters.

10.3.6.3.1.1. BlockAddID Element

name: BlockAddID

path: "\Segment\Cluster\BlockGroup\BlockAdditions\BlockMore\BlockAddID"

id: 0xEE

minOccurs: 1

maxOccurs: 1

range: not 0

default: 1

type: uinteger

definition: An ID to identify the BlockAdditional level. If BlockAddIDType of the corresponding block is 0, this value is also the value of BlockAddIDType for the meaning of the content of BlockAdditional.

10.3.6.3.1.2. BlockAdditional Element

name: BlockAdditional

path: "\Segment\Cluster\BlockGroup\BlockAdditions\BlockMore\BlockAdditional"

id: 0xA5

minOccurs: 1

maxOccurs: 1

type: binary

definition: Interpreted by the codec as it wishes (using the BlockAddID).

10.3.6.4. BlockDuration Element

name: BlockDuration

path: "\Segment\Cluster\BlockGroup\BlockDuration"

id: 0x9B

minOccurs: see implementation notes

maxOccurs: 1

default: see implementation notes

type: uinteger

definition: The duration of the Block (based on TimestampScale). The BlockDuration Element can be useful at the end of a Track to define the duration of the last frame (as there is no subsequent Block available), or when there is a break in a track like for subtitle tracks.

implementation notes:

attribute	note
minOccurs	BlockDuration MUST be set (minOccurs=1) if the associated TrackEntry stores a DefaultDuration value.
default	When not written and with no DefaultDuration, the value is assumed to be the difference between the timestamp of this Block and the timestamp of the next Block in "display" order (not coding order).

Table 11

10.3.6.5. ReferencePriority Element

name: ReferencePriority

path: "\Segment\Cluster\BlockGroup\ReferencePriority"

id: 0xFA

minOccurs: 1

maxOccurs: 1

default: 0

type: uinteger

definition: This frame is referenced and has the specified cache priority. In cache only a frame of the same or higher priority can replace this frame. A value of 0 means the frame is not referenced.

10.3.6.6. ReferenceBlock Element

name: ReferenceBlock

path: "\Segment\Cluster\BlockGroup\ReferenceBlock"

id: 0xFB

type: integer

definition: Timestamp of another frame used as a reference (ie: B or P frame). The timestamp is relative to the block it's attached to.

10.3.6.7. ReferenceVirtual Element

name: ReferenceVirtual

path: "\Segment\Cluster\BlockGroup\ReferenceVirtual"

id: 0xFD

maxOccurs: 1

type: integer

minver: 0

maxver: 0

definition: The Segment Position of the data that would otherwise be in position of the virtual block.

10.3.6.8. CodecState Element

name: CodecState

path: "\Segment\Cluster\BlockGroup\CodecState"

id: 0xA4

maxOccurs: 1

type: binary

minver: 2

definition: The new codec state to use. Data interpretation is private to the codec. This information SHOULD always be referenced by a seek entry.

10.3.6.9. DiscardPadding Element

name: DiscardPadding

path: "\Segment\Cluster\BlockGroup\DiscardPadding"

id: 0x75A2

maxOccurs: 1

type: integer

minver: 4

definition: Duration in nanoseconds of the silent data added to the Block (padding at the end of the Block for positive value, at the beginning of the Block for negative value). The duration of DiscardPadding is not calculated in the duration of the TrackEntry and SHOULD be discarded during playback.

10.3.6.10. Slices Element

name: Slices

path: "\Segment\Cluster\BlockGroup\Slices"

id: 0x8E

maxOccurs: 1

type: master

definition: Contains slices description.

10.3.6.10.1. TimeSlice Element

name: TimeSlice

path: "\Segment\Cluster\BlockGroup\Slices\TimeSlice"

id: 0xE8

type: master

maxver: 1

definition: Contains extra time information about the data contained

in the Block. Being able to interpret this Element is not REQUIRED for playback.

10.3.6.10.1.1. LaceNumber Element

name: LaceNumber

path: "\Segment\Cluster\BlockGroup\Slices\TimeSlice\LaceNumber"

id: 0xCC

maxOccurs: 1

default: 0

type: uinteger

maxver: 1

definition: The reverse number of the frame in the lace (0 is the last frame, 1 is the next to last, etc). Being able to interpret this Element is not REQUIRED for playback.

10.3.6.10.1.2. FrameNumber Element

name: FrameNumber

path: "\Segment\Cluster\BlockGroup\Slices\TimeSlice\FrameNumber"

id: 0xCD

maxOccurs: 1

default: 0

type: uinteger

minver: 0

maxver: 0

definition: The number of the frame to generate from this lace with this delay (allow you to generate many frames from the same Block/Frame).

10.3.6.10.1.3. BlockAdditionID Element

name: BlockAdditionID

path: "\Segment\Cluster\BlockGroup\Slices\TimeSlice\BlockAdditionID"

id: 0xCB

maxOccurs: 1

default: 0

type: uinteger

minver: 0

maxver: 0

definition: The ID of the BlockAdditional Element (0 is the main Block).

10.3.6.10.1.4. Delay Element

name: Delay

path: "\Segment\Cluster\BlockGroup\Slices\TimeSlice\Delay"

id: 0xCE

maxOccurs: 1

default: 0

type: uinteger

minver: 0

maxver: 0

definition: The (scaled) delay to apply to the Element.

10.3.6.10.1.5. SliceDuration Element

name: SliceDuration

path: "\Segment\Cluster\BlockGroup\Slices\TimeSlice\SliceDuration"

id: 0xCF

maxOccurs: 1

default: 0

type: uinteger

minver: 0

maxver: 0

definition: The (scaled) duration to apply to the Element.

10.3.6.11. ReferenceFrame Element

name: ReferenceFrame

path: "\Segment\Cluster\BlockGroup\ReferenceFrame"

id: 0xC8

maxOccurs: 1

type: master

minver: 0

maxver: 0

definition: DivX trick track extensions

10.3.6.11.1. ReferenceOffset Element

name: ReferenceOffset

path: "\Segment\Cluster\BlockGroup\ReferenceFrame\ReferenceOffset"

id: 0xC9

minOccurs: 1

maxOccurs: 1

type: uinteger

minver: 0

maxver: 0

definition: DivX trick track extensions

10.3.6.11.2. ReferenceTimestamp Element

name: ReferenceTimestamp

path: "\Segment\Cluster\BlockGroup\ReferenceFrame\ReferenceTimestamp"

id: 0xCA

minOccurs: 1

maxOccurs: 1

type: uinteger

minver: 0

maxver: 0

definition: DivX trick track extensions

10.3.7. EncryptedBlock Element

name: EncryptedBlock

path: "\Segment\Cluster\EncryptedBlock"

id: 0xAF

type: binary

minver: 0

maxver: 0

definition: Similar to SimpleBlock, see Section 6.2.4.4, but the data inside the Block are Transformed (encrypt and/or signed).

10.4. Tracks Element

name: Tracks

path: "\Segment\Tracks"

id: 0x1654AE6B

type: master

recurring: 1

definition: A Top-Level Element of information with many tracks described.

10.4.1. TrackEntry Element

name: TrackEntry

path: "\Segment\Tracks\TrackEntry"

id: 0xAE

minOccurs: 1

type: master

definition: Describes a track with all Elements.

10.4.1.1. TrackNumber Element

name: TrackNumber

path: "\Segment\Tracks\TrackEntry\TrackNumber"

id: 0xD7

minOccurs: 1

maxOccurs: 1

range: not 0

type: uinteger

definition: The track number as used in the Block Header (using more than 127 tracks is not encouraged, though the design allows an unlimited number).

10.4.1.2. TrackUID Element

name: TrackUID

path: "\Segment\Tracks\TrackEntry\TrackUID"

id: 0x73C5

minOccurs: 1

maxOccurs: 1

range: not 0

type: uinteger

definition: A unique ID to identify the Track. This SHOULD be kept the same when making a direct stream copy of the Track to another file.

10.4.1.3. TrackType Element

name: TrackType

path: "\Segment\Tracks\TrackEntry\TrackType"

id: 0x83

minOccurs: 1

maxOccurs: 1

range: 1-254

type: uinteger

definition: A set of track types coded on 8 bits.

restrictions:

value	label
1	video
2	audio
3	complex
16	logo
17	subtitle
18	buttons
32	control
33	metadata

Table 12

10.4.1.4. FlagEnabled Element

name: FlagEnabled

path: "\Segment\Tracks\TrackEntry\FlagEnabled"

id: 0xB9

minOccurs: 1

maxOccurs: 1

range: 0-1

default: 1

type: uinteger

minver: 2

definition: Set if the track is usable. (1 bit)

10.4.1.5. FlagDefault Element

name: FlagDefault

path: "\Segment\Tracks\TrackEntry\FlagDefault"

id: 0x88

minOccurs: 1

maxOccurs: 1

range: 0-1

default: 1

type: uinteger

definition: Set if that track (audio, video or subs) SHOULD be active if no language found matches the user preference. (1 bit)

10.4.1.6. FlagForced Element

name: FlagForced

path: "\Segment\Tracks\TrackEntry\FlagForced"

id: 0x55AA

minOccurs: 1

maxOccurs: 1

range: 0-1

default: 0

type: uinteger

definition: Set if that track MUST be active during playback. There can be many forced track for a kind (audio, video or subs), the player SHOULD select the one which language matches the user preference or the default + forced track. Overlay MAY happen between a forced and non-forced track of the same kind. (1 bit)

10.4.1.7. FlagLacing Element

name: FlagLacing

path: "\Segment\Tracks\TrackEntry\FlagLacing"

id: 0x9C

minOccurs: 1

maxOccurs: 1

range: 0-1

default: 1

type: uinteger

definition: Set if the track MAY contain blocks using lacing. (1 bit)

10.4.1.8. MinCache Element

name: MinCache

path: "\Segment\Tracks\TrackEntry\MinCache"

id: 0x6DE7

minOccurs: 1

maxOccurs: 1

default: 0

type: uinteger

definition: The minimum number of frames a player SHOULD be able to cache during playback. If set to 0, the reference pseudo-cache system is not used.

10.4.1.9. MaxCache Element

name: MaxCache

path: "\Segment\Tracks\TrackEntry\MaxCache"

id: 0x6DF8

maxOccurs: 1

type: uinteger

definition: The maximum cache size necessary to store referenced frames in and the current frame. 0 means no cache is needed.

10.4.1.10. DefaultDuration Element

name: DefaultDuration

path: "\Segment\Tracks\TrackEntry\DefaultDuration"

id: 0x23E383

maxOccurs: 1

range: not 0

type: uinteger

definition: Number of nanoseconds (not scaled via TimestampScale) per frame (frame in the Matroska sense -- one Element put into a (Simple)Block).

10.4.1.11. DefaultDecodedFieldDuration Element

name: DefaultDecodedFieldDuration

path: "\Segment\Tracks\TrackEntry\DefaultDecodedFieldDuration"

id: 0x234E7A

maxOccurs: 1

range: not 0

type: uinteger

minver: 4

definition: The period in nanoseconds (not scaled by TimestampScale) between two successive fields at the output of the decoding process, see Section 19 for more information

10.4.1.12. TrackTimestampScale Element

name: TrackTimestampScale

path: "\Segment\Tracks\TrackEntry\TrackTimestampScale"

id: 0x23314F

minOccurs: 1

maxOccurs: 1

range: > 0x0p+0

default: 0x1p+0

type: float

maxver: 3

definition: DEPRECATED, DO NOT USE. The scale to apply on this track to work at normal speed in relation with other tracks (mostly used to adjust video speed when the audio length differs).

10.4.1.13. TrackOffset Element

name: TrackOffset

path: "\Segment\Tracks\TrackEntry\TrackOffset"

id: 0x537F

maxOccurs: 1

default: 0

type: integer

minver: 0

maxver: 0

definition: A value to add to the Block's Timestamp. This can be used to adjust the playback offset of a track.

10.4.1.14. MaxBlockAdditionID Element

name: MaxBlockAdditionID

path: "\Segment\Tracks\TrackEntry\MaxBlockAdditionID"

id: 0x55EE

minOccurs: 1

maxOccurs: 1

default: 0

type: uinteger

definition: The maximum value of BlockAddID (Section 10.3.6.3.1.1). A value 0 means there is no BlockAdditions (Section 10.3.6.3) for this track.

10.4.1.15. BlockAdditionMapping Element

name: BlockAdditionMapping

path: "\Segment\Tracks\TrackEntry\BlockAdditionMapping"

id: 0x41E4

type: master

minver: 4

definition: Contains elements that extend the track format, by adding content either to each frame, with BlockAddID (Section 10.3.6.3.1.1), or to the track as a whole with BlockAddIDExtraData.

10.4.1.15.1. BlockAddIDValue Element

name: BlockAddIDValue

path: "\Segment\Tracks\TrackEntry\BlockAdditionMapping\BlockAddIDValue"

id: 0x41F0

maxOccurs: 1

range: >=2

type: uinteger

minver: 4

definition: If the track format extension needs content beside frames, the value refers to the BlockAddID (Section 10.3.6.3.1.1), value being described. To keep MaxBlockAdditionID as low as possible, small values SHOULD be used.

10.4.1.15.2. BlockAddIDName Element

name: BlockAddIDName

path: "\Segment\Tracks\TrackEntry\BlockAdditionMapping\BlockAddIDName"

id: 0x41A4

maxOccurs: 1

type: string

minver: 4

definition: A human-friendly name describing the type of BlockAdditional data, as defined by the associated Block Additional Mapping.

10.4.1.15.3. BlockAddIDType Element

name: BlockAddIDType

path: "\Segment\Tracks\TrackEntry\BlockAdditionMapping\BlockAddIDType"

id: 0x41E7

minOccurs: 1

maxOccurs: 1

default: 0

type: uinteger

minver: 4

definition: Stores the registered identifer of the Block Additional Mapping to define how the BlockAdditional data should be handled.

10.4.1.15.4. BlockAddIDExtraData Element

name: BlockAddIDExtraData

path: "\Segment\Tracks\TrackEntry\BlockAdditionMapping\BlockAddIDExtraData"

id: 0x41ED

maxOccurs: 1

type: binary

minver: 4

definition: Extra binary data that the BlockAddIDType can use to interpret the BlockAdditional data. The intepretation of the binary data depends on the BlockAddIDType value and the corresponding Block Additional Mapping.

10.4.1.16. Name Element

name: Name

path: "\Segment\Tracks\TrackEntry\Name"

id: 0x536E

maxOccurs: 1

type: utf-8

definition: A human-readable track name.

10.4.1.17. Language Element

name: Language

path: "\Segment\Tracks\TrackEntry\Language"

id: 0x22B59C

maxOccurs: 1

default: eng

type: string

definition: Specifies the language of the track in the Matroska languages form; see Section 6.2.1 on language codes. This Element MUST be ignored if the LanguageIETF Element is used in the same TrackEntry.

10.4.1.18. LanguageIETF Element

name: LanguageIETF

path: "\Segment\Tracks\TrackEntry\LanguageIETF"

id: 0x22B59D

maxOccurs: 1

type: string

minver: 4

definition: Specifies the language of the track according to BCP 47 and using the IANA Language Subtag Registry. If this Element is used, then any Language Elements used in the same TrackEntry MUST be ignored.

10.4.1.19. CodecID Element

name: CodecID

path: "\Segment\Tracks\TrackEntry\CodecID"

id: 0x86

minOccurs: 1

maxOccurs: 1

type: string

definition: An ID corresponding to the codec, see [I-D.ietf-cellar-codec] for more info.

10.4.1.20. CodecPrivate Element

name: CodecPrivate

path: "\Segment\Tracks\TrackEntry\CodecPrivate"

id: 0x63A2

maxOccurs: 1

type: binary

definition: Private data only known to the codec.

10.4.1.21. CodecName Element

name: CodecName

path: "\Segment\Tracks\TrackEntry\CodecName"

id: 0x258688

maxOccurs: 1

type: utf-8

definition: A human-readable string specifying the codec.

10.4.1.22. AttachmentLink Element

name: AttachmentLink

path: "\Segment\Tracks\TrackEntry\AttachmentLink"

id: 0x7446

maxOccurs: 1

range: not 0

type: uinteger

maxver: 3

definition: The UID of an attachment that is used by this codec.

10.4.1.23. CodecSettings Element

name: CodecSettings

path: "\Segment\Tracks\TrackEntry\CodecSettings"

id: 0x3A9697

maxOccurs: 1

type: utf-8

minver: 0

maxver: 0

definition: A string describing the encoding setting used.

10.4.1.24. CodecInfoURL Element

name: CodecInfoURL

path: "\Segment\Tracks\TrackEntry\CodecInfoURL"

id: 0x3B4040

type: string

minver: 0

maxver: 0

definition: A URL to find information about the codec used.

10.4.1.25. CodecDownloadURL Element

name: CodecDownloadURL

path: "\Segment\Tracks\TrackEntry\CodecDownloadURL"

id: 0x26B240

type: string

minver: 0

maxver: 0

definition: A URL to download about the codec used.

10.4.1.26. CodecDecodeAll Element

name: CodecDecodeAll

path: "\Segment\Tracks\TrackEntry\CodecDecodeAll"

id: 0xAA

minOccurs: 1

maxOccurs: 1

range: 0-1

default: 1

type: uinteger

minver: 2

definition: The codec can decode potentially damaged data (1 bit).

10.4.1.27. TrackOverlay Element

name: TrackOverlay

path: "\Segment\Tracks\TrackEntry\TrackOverlay"

id: 0x6FAB

type: uinteger

definition: Specify that this track is an overlay track for the Track specified (in the u-integer). That means when this track has a gap, see Section 10.3.2 on SilentTracks, the overlay track SHOULD be used instead. The order of multiple TrackOverlay matters, the first one is the one that SHOULD be used. If not found it SHOULD be the second, etc.

10.4.1.28. CodecDelay Element

name: CodecDelay

path: "\Segment\Tracks\TrackEntry\CodecDelay"

id: 0x56AA

maxOccurs: 1

default: 0

type: uinteger

minver: 4

definition: CodecDelay is The codec-built-in delay in nanoseconds. This value MUST be subtracted from each block timestamp in order to get the actual timestamp. The value SHOULD be small so the muxing of tracks with the same actual timestamp are in the same Cluster.

10.4.1.29. SeekPreRoll Element

name: SeekPreRoll

path: "\Segment\Tracks\TrackEntry\SeekPreRoll"

id: 0x56BB

minOccurs: 1

maxOccurs: 1

default: 0

type: uinteger

minver: 4

definition: After a discontinuity, SeekPreRoll is the duration in nanoseconds of the data the decoder MUST decode before the decoded data is valid.

10.4.1.30. TrackTranslate Element

name: TrackTranslate

path: "\Segment\Tracks\TrackEntry\TrackTranslate"

id: 0x6624

type: master

definition: The track identification for the given Chapter Codec.

10.4.1.30.1. TrackTranslateEditionUID Element

name: TrackTranslateEditionUID

path: "\Segment\Tracks\TrackEntry\TrackTranslate\TrackTranslateEditionUID"

id: 0x66FC

type: uinteger

definition: Specify an edition UID on which this translation applies. When not specified, it means for all editions found in the Segment.

10.4.1.30.2. TrackTranslateCodec Element

name: TrackTranslateCodec

path: "\Segment\Tracks\TrackEntry\TrackTranslate\TrackTranslateCodec"

id: 0x66BF

minOccurs: 1

maxOccurs: 1

type: uinteger

definition: The chapter codec; see Section 10.7.1.5.12.1.

restrictions:

value	label
0	Matroska Script
1	DVD-menu

Table 13

10.4.1.30.3. TrackTranslateTrackID Element

name: TrackTranslateTrackID

path: "\Segment\Tracks\TrackEntry\TrackTranslate\TrackTranslateTrackID"

id: 0x66A5

minOccurs: 1

maxOccurs: 1

type: binary

definition: The binary value used to represent this track in the chapter codec data. The format depends on the ChapProcessCodecID used; see Section 10.7.1.5.12.1.

10.4.1.31. Video Element

name: Video

path: "\Segment\Tracks\TrackEntry\Video"

id: 0xE0

maxOccurs: 1

type: master

definition: Video settings.

10.4.1.31.1. FlagInterlaced Element

name: FlagInterlaced

path: "\Segment\Tracks\TrackEntry\Video\FlagInterlaced"

id: 0x9A

minOccurs: 1

maxOccurs: 1

range: 0-2

default: 0

type: uinteger

minver: 2

definition: A flag to declare if the video is known to be progressive, or interlaced, and if applicable to declare details about the interlacement.

restrictions:

value	label
0	undetermined
1	interlaced
2	progressive

Table 14

10.4.1.31.2. FieldOrder Element

name: FieldOrder

path: "\Segment\Tracks\TrackEntry\Video\FieldOrder"

id: 0x9D

minOccurs: 1

maxOccurs: 1

range: 0-14

default: 2

type: uinteger

minver: 4

definition: Declare the field ordering of the video. If FlagInterlaced is not set to 1, this Element MUST be ignored.

restrictions:

value	label	documentation
0	progressive	
1	tff	Top field displayed first. Top field stored first.
2	undetermined	
6	bff	Bottom field displayed first. Bottom field stored first.
9	bff(swapped)	Top field displayed first. Fields are interleaved in storage with the top line of the top field stored first.
14	tff(swapped)	Bottom field displayed first. Fields are interleaved in storage with the top line of the top field stored first.

Table 15

10.4.1.31.3. StereoMode Element

name: StereoMode

path: "\Segment\Tracks\TrackEntry\Video\StereoMode"

id: 0x53B8

maxOccurs: 1

default: 0

type: uinteger

minver: 3

definition: Stereo-3D video mode. There are some more details in Section 26.5.

restrictions:

value	label
0	mono
1	side by side (left eye first)
2	top - bottom (right eye is first)
3	top - bottom (left eye is first)
4	checkboard (right eye is first)
5	checkboard (left eye is first)
6	row interleaved (right eye is first)
7	row interleaved (left eye is first)
8	column interleaved (right eye is first)
9	column interleaved (left eye is first)
10	anaglyph (cyan/red)
11	side by side (right eye first)
12	anaglyph (green/magenta)
13	both eyes laced in one Block (left eye is first)
14	both eyes laced in one Block (right eye is first)

Table 16

10.4.1.31.4. AlphaMode Element

name: AlphaMode

path: "\Segment\Tracks\TrackEntry\Video\AlphaMode"

id: 0x53C0

maxOccurs: 1

default: 0

type: uinteger

minver: 3

definition: Alpha Video Mode. Presence of this Element indicates that the BlockAdditional Element could contain Alpha data.

10.4.1.31.5. OldStereoMode Element

name: OldStereoMode

path: "\Segment\Tracks\TrackEntry\Video\OldStereoMode"

id: 0x53B9

maxOccurs: 1

type: uinteger

maxver: 0

definition: DEPRECATED, DO NOT USE. Bogus StereoMode value used in old versions of libmatroska.

restrictions:

value	label
0	mono
1	right eye
2	left eye
3	both eyes

Table 17

10.4.1.31.6. PixelWidth Element

name: PixelWidth

path: "\Segment\Tracks\TrackEntry\Video\PixelWidth"

id: 0xB0

minOccurs: 1

maxOccurs: 1

range: not 0

type: uinteger

definition: Width of the encoded video frames in pixels.

10.4.1.31.7. PixelHeight Element

name: PixelHeight

path: "\Segment\Tracks\TrackEntry\Video\PixelHeight"

id: 0xBA

minOccurs: 1

maxOccurs: 1

range: not 0

type: uinteger

definition: Height of the encoded video frames in pixels.

10.4.1.31.8. PixelCropBottom Element

name: PixelCropBottom

path: "\Segment\Tracks\TrackEntry\Video\PixelCropBottom"

id: 0x54AA

maxOccurs: 1

default: 0

type: uinteger

definition: The number of video pixels to remove at the bottom of the image.

10.4.1.31.9. PixelCropTop Element

name: PixelCropTop

path: "\Segment\Tracks\TrackEntry\Video\PixelCropTop"

id: 0x54BB

maxOccurs: 1

default: 0

type: uinteger

definition: The number of video pixels to remove at the top of the image.

10.4.1.31.10. PixelCropLeft Element

name: PixelCropLeft

path: "\Segment\Tracks\TrackEntry\Video\PixelCropLeft"

id: 0x54CC

maxOccurs: 1

default: 0

type: uinteger

definition: The number of video pixels to remove on the left of the image.

10.4.1.31.11. PixelCropRight Element

name: PixelCropRight

path: "\Segment\Tracks\TrackEntry\Video\PixelCropRight"

id: 0x54DD

maxOccurs: 1

default: 0

type: uinteger

definition: The number of video pixels to remove on the right of the image.

10.4.1.31.12. DisplayWidth Element

name: DisplayWidth

path: "\Segment\Tracks\TrackEntry\Video\DisplayWidth"

id: 0x54B0

maxOccurs: 1

range: not 0

default: see implementation notes

type: uinteger

definition: Width of the video frames to display. Applies to the video frame after cropping (PixelCrop* Elements).

implementation notes:

attribute	note
default	If the DisplayUnit of the same TrackEntry is 0, then the default value for DisplayWidth is equal to
PixelWidth - PixelCropLeft - PixelCropRight, else there is no default value.	

Table 18

10.4.1.31.13. DisplayHeight Element

name: DisplayHeight

path: "\Segment\Tracks\TrackEntry\Video\DisplayHeight"

id: 0x54BA

maxOccurs: 1

range: not 0

default: see implementation notes

type: uinteger

definition: Height of the video frames to display. Applies to the video frame after cropping (PixelCrop* Elements).

implementation notes:

attribute	note
default	If the DisplayUnit of the same TrackEntry is 0, then the default value for DisplayHeight is equal to
PixelHeight - PixelCropTop - PixelCropBottom, else there is no default value.	

Table 19

10.4.1.31.14. DisplayUnit Element

name: DisplayUnit

path: "\Segment\Tracks\TrackEntry\Video\DisplayUnit"

id: 0x54B2

maxOccurs: 1

default: 0

type: uinteger

definition: How DisplayWidth & DisplayHeight are interpreted.

restrictions:

value	label
0	pixels
1	centimeters
2	inches
3	display aspect ratio
4	unknown

Table 20

10.4.1.31.15. AspectRatioType Element

name: AspectRatioType

path: "\Segment\Tracks\TrackEntry\Video\AspectRatioType"

id: 0x54B3

maxOccurs: 1

default: 0

type: uinteger

definition: Specify the possible modifications to the aspect ratio.

restrictions:

value	label
0	free resizing
1	keep aspect ratio
2	fixed

Table 21

10.4.1.31.16. ColourSpace Element

name: ColourSpace

path: "\Segment\Tracks\TrackEntry\Video\ColourSpace"

id: 0x2EB524

minOccurs: see implementation notes

maxOccurs: 1

type: binary

definition: Specify the pixel format used for the Track's data as a FourCC. This value is similar in scope to the biCompression value of AVI's BITMAPINFOHEADER.

implementation notes:

attribute	note
minOccurs	ColourSpace MUST be set (minOccurs=1) in TrackEntry, when the CodecID Element of the TrackEntry is set to "V_UNCOMPRESSED".

Table 22

10.4.1.31.17. GammaValue Element

name: GammaValue

path: "\Segment\Tracks\TrackEntry\Video\GammaValue"

id: 0x2FB523

maxOccurs: 1

range: > 0x0p+0

type: float

minver: 0

maxver: 0

definition: Gamma Value.

10.4.1.31.18. FrameRate Element

name: FrameRate

path: "\Segment\Tracks\TrackEntry\Video\FrameRate"

id: 0x2383E3

maxOccurs: 1

range: > 0x0p+0

type: float

minver: 0

maxver: 0

definition: Number of frames per second. Informational only.

10.4.1.31.19. Colour Element

name: Colour

path: "\Segment\Tracks\TrackEntry\Video\Colour"

id: 0x55B0

maxOccurs: 1

type: master

minver: 4

definition: Settings describing the colour format.

10.4.1.31.19.1. MatrixCoefficients Element

name: MatrixCoefficients

path: "\Segment\Tracks\TrackEntry\Video\Colour\MatrixCoefficients"

id: 0x55B1

maxOccurs: 1

default: 2

type: uinteger

minver: 4

definition: The Matrix Coefficients of the video used to derive luma and chroma values from red, green, and blue color primaries. For clarity, the value and meanings for MatrixCoefficients are adopted from Table 4 of ISO/IEC 23001-8:2016 or ITU-T H.273.

restrictions:

value	label
0	Identity
1	ITU-R BT.709
2	unspecified
3	reserved
4	US FCC 73.682
5	ITU-R BT.470BG
6	SMPTE 170M
7	SMPTE 240M
8	YCoCg
9	BT2020 Non-constant Luminance
10	BT2020 Constant Luminance
11	SMPTE ST 2085
12	Chroma-derived Non-constant Luminance
13	Chroma-derived Constant Luminance
14	ITU-R BT.2100-0

Table 23

10.4.1.31.19.2. BitsPerChannel Element

name: BitsPerChannel

path: "\Segment\Tracks\TrackEntry\Video\Colour\BitsPerChannel"

id: 0x55B2

maxOccurs: 1

default: 0

type: uinteger

minver: 4

definition: Number of decoded bits per channel. A value of 0 indicates that the BitsPerChannel is unspecified.

10.4.1.31.19.3. ChromaSubsamplingHorz Element

name: ChromaSubsamplingHorz

path: "\Segment\Tracks\TrackEntry\Video\Colour\ChromaSubsamplingHorz"

id: 0x55B3

maxOccurs: 1

type: uinteger

minver: 4

definition: The amount of pixels to remove in the Cr and Cb channels for every pixel not removed horizontally. Example: For video with 4:2:0 chroma subsampling, the ChromaSubsamplingHorz SHOULD be set to 1.

10.4.1.31.19.4. ChromaSubsamplingVert Element

name: ChromaSubsamplingVert

path: "\Segment\Tracks\TrackEntry\Video\Colour\ChromaSubsamplingVert"

id: 0x55B4

maxOccurs: 1

type: uinteger

minver: 4

definition: The amount of pixels to remove in the Cr and Cb channels for every pixel not removed vertically. Example: For video with 4:2:0 chroma subsampling, the ChromaSubsamplingVert SHOULD be set to 1.

10.4.1.31.19.5. CbSubsamplingHorz Element

name: CbSubsamplingHorz

path: "\Segment\Tracks\TrackEntry\Video\Colour\CbSubsamplingHorz"

id: 0x55B5

maxOccurs: 1

type: uinteger

minver: 4

definition: The amount of pixels to remove in the Cb channel for every pixel not removed horizontally. This is additive with ChromaSubsamplingHorz. Example: For video with 4:2:1 chroma subsampling, the ChromaSubsamplingHorz SHOULD be set to 1 and CbSubsamplingHorz SHOULD be set to 1.

10.4.1.31.19.6. CbSubsamplingVert Element

name: CbSubsamplingVert

path: "\Segment\Tracks\TrackEntry\Video\Colour\CbSubsamplingVert"

id: 0x55B6

maxOccurs: 1

type: uinteger

minver: 4

definition: The amount of pixels to remove in the Cb channel for every pixel not removed vertically. This is additive with ChromaSubsamplingVert.

10.4.1.31.19.7. ChromaSitingHorz Element

name: ChromaSitingHorz

path: "\Segment\Tracks\TrackEntry\Video\Colour\ChromaSitingHorz"

id: 0x55B7

maxOccurs: 1

default: 0

type: uinteger

minver: 4

definition: How chroma is subsampled horizontally.

restrictions:

value	label
0	unspecified
1	left collocated
2	half

Table 24

10.4.1.31.19.8. ChromaSitingVert Element

name: ChromaSitingVert

path: "\Segment\Tracks\TrackEntry\Video\Colour\ChromaSitingVert"

id: 0x55B8

maxOccurs: 1

default: 0

type: uinteger

minver: 4

definition: How chroma is subsampled vertically.

restrictions:

value	label
0	unspecified
1	top collocated
2	half

Table 25

10.4.1.31.19.9. Range Element

name: Range

path: "\Segment\Tracks\TrackEntry\Video\Colour\Range"

id: 0x55B9

maxOccurs: 1

default: 0

type: uinteger

minver: 4

definition: Clipping of the color ranges.

restrictions:

value	label
0	unspecified
1	broadcast range
2	full range (no clipping)
3	defined by MatrixCoefficients / TransferCharacteristics

Table 26

10.4.1.31.19.10. TransferCharacteristics Element

name: TransferCharacteristics

path: "\Segment\Tracks\TrackEntry\Video\Colour\TransferCharacteristics"

id: 0x55BA

maxOccurs: 1

default: 2

type: uinteger

minver: 4

definition: The transfer characteristics of the video. For clarity, the value and meanings for TransferCharacteristics are adopted from Table 3 of ISO/IEC 23091-4 or ITU-T H.273.

restrictions:

value	label
0	reserved
1	ITU-R BT.709
2	unspecified
3	reserved
4	Gamma 2.2 curve - BT.470M
5	Gamma 2.8 curve - BT.470BG
6	SMPTE 170M
7	SMPTE 240M
8	Linear
9	Log
10	Log Sqrt
11	IEC 61966-2-4
12	ITU-R BT.1361 Extended Colour Gamut
13	IEC 61966-2-1
14	ITU-R BT.2020 10 bit
15	ITU-R BT.2020 12 bit
16	ITU-R BT.2100 Perceptual Quantization
17	SMPTE ST 428-1
18	ARIB STD-B67 (HLG)

Table 27

10.4.1.31.19.11. Primaries Element

name: Primaries

path: "\Segment\Tracks\TrackEntry\Video\Colour\Primaries"

id: 0x55BB

maxOccurs: 1

default: 2

type: uinteger

minver: 4

definition: The colour primaries of the video. For clarity, the value and meanings for Primaries are adopted from Table 2 of ISO/IEC 23091-4 or ITU-T H.273.

restrictions:

value	label
0	reserved
1	ITU-R BT.709
2	unspecified
3	reserved
4	ITU-R BT.470M
5	ITU-R BT.470BG - BT.601 625
6	ITU-R BT.601 525 - SMPTE 170M
7	SMPTE 240M
8	FILM
9	ITU-R BT.2020
10	SMPTE ST 428-1
11	SMPTE RP 432-2
12	SMPTE EG 432-2
22	EBU Tech. 3213-E - JEDEC P22 phosphors

Table 28

10.4.1.31.19.12. MaxCLL Element

name: MaxCLL

path: "\Segment\Tracks\TrackEntry\Video\Colour\MaxCLL"

id: 0x55BC

maxOccurs: 1

type: uinteger

minver: 4

definition: Maximum brightness of a single pixel (Maximum Content Light Level) in candelas per square meter (cd/m²).

10.4.1.31.19.13. MaxFALL Element

name: MaxFALL

path: "\Segment\Tracks\TrackEntry\Video\Colour\MaxFALL"

id: 0x55BD

maxOccurs: 1

type: uinteger

minver: 4

definition: Maximum brightness of a single full frame (Maximum Frame-Average Light Level) in candelas per square meter (cd/m²).

10.4.1.31.19.14. MasteringMetadata Element

name: MasteringMetadata

path: "\Segment\Tracks\TrackEntry\Video\Colour\MasteringMetadata"

id: 0x55D0

maxOccurs: 1

type: master

minver: 4

definition: SMPTE 2086 mastering data.

10.4.1.31.19.15. PrimaryRChromaticityX Element

name: PrimaryRChromaticityX

path: "\Segment\Tracks\TrackEntry\Video\Colour\MasteringMetadata\PrimaryRChromaticityX"

id: 0x55D1

maxOccurs: 1

range: 0-1

type: float

minver: 4

definition: Red X chromaticity coordinate, as defined by CIE 1931.

10.4.1.31.19.16. PrimaryRChromaticityY Element

name: PrimaryRChromaticityY

path: "\Segment\Tracks\TrackEntry\Video\Colour\MasteringMetadata\PrimaryRChromaticityY"

id: 0x55D2

maxOccurs: 1

range: 0-1

type: float

minver: 4

definition: Red Y chromaticity coordinate, as defined by CIE 1931.

10.4.1.31.19.17. PrimaryGChromaticityX Element

name: PrimaryGChromaticityX

path: "\Segment\Tracks\TrackEntry\Video\Colour\MasteringMetadata\PrimaryGChromaticityX"

id: 0x55D3

maxOccurs: 1

range: 0-1

type: float

minver: 4

definition: Green X chromaticity coordinate, as defined by CIE 1931.

10.4.1.31.19.18. PrimaryGChromaticityY Element

name: PrimaryGChromaticityY

path: "\Segment\Tracks\TrackEntry\Video\Colour\MasteringMetadata\PrimaryGChromaticityY"

id: 0x55D4

maxOccurs: 1

range: 0-1

type: float

minver: 4

definition: Green Y chromaticity coordinate, as defined by CIE 1931.

10.4.1.31.19.19. PrimaryBChromaticityX Element

name: PrimaryBChromaticityX

path: "\Segment\Tracks\TrackEntry\Video\Colour\MasteringMetadata\PrimaryBChromaticityX"

id: 0x55D5

maxOccurs: 1

range: 0-1

type: float

minver: 4

definition: Blue X chromaticity coordinate, as defined by CIE 1931.

10.4.1.31.19.20. PrimaryBChromaticityY Element

name: PrimaryBChromaticityY

path: "\Segment\Tracks\TrackEntry\Video\Colour\MasteringMetadata\PrimaryBChromaticityY"

id: 0x55D6

maxOccurs: 1

range: 0-1

type: float

minver: 4

definition: Blue Y chromaticity coordinate, as defined by CIE 1931.

10.4.1.31.19.21. WhitePointChromaticityX Element

name: WhitePointChromaticityX

path: "\Segment\Tracks\TrackEntry\Video\Colour\MasteringMetadata\WhitePointChromaticityX"

id: 0x55D7

maxOccurs: 1

range: 0-1

type: float

minver: 4

definition: White X chromaticity coordinate, as defined by CIE 1931.

10.4.1.31.19.22. WhitePointChromaticityY Element

name: WhitePointChromaticityY

path: "\Segment\Tracks\TrackEntry\Video\Colour\MasteringMetadata\WhitePointChromaticityY"

id: 0x55D8

maxOccurs: 1

range: 0-1

type: float

minver: 4

definition: White Y chromaticity coordinate, as defined by CIE 1931.

10.4.1.31.19.23. LuminanceMax Element

name: LuminanceMax

path: "\Segment\Tracks\TrackEntry\Video\Colour\MasteringMetadata\LuminanceMax"

id: 0x55D9

maxOccurs: 1

range: >= 0x0p+0

type: float

minver: 4

definition: Maximum luminance. Represented in candelas per square meter (cd/m²).

10.4.1.31.19.24. LuminanceMin Element

name: LuminanceMin

path: "\Segment\Tracks\TrackEntry\Video\Colour\MasteringMetadata\LuminanceMin"

id: 0x55DA

maxOccurs: 1

range: >= 0x0p+0

type: float

minver: 4

definition: Minimum luminance. Represented in candelas per square meter (cd/m²).

10.4.1.31.20. Projection Element

name: Projection

path: "\Segment\Tracks\TrackEntry\Video\Projection"

id: 0x7670

maxOccurs: 1

type: master

minver: 4

definition: Describes the video projection details. Used to render spherical and VR videos.

10.4.1.31.20.1. ProjectionType Element

name: ProjectionType

path: "\Segment\Tracks\TrackEntry\Video\Projection\ProjectionType"

id: 0x7671

minOccurs: 1

maxOccurs: 1

range: 0-3

default: 0

type: uinteger

minver: 4

definition: Describes the projection used for this video track.

restrictions:

value	label
0	rectangular
1	equirectangular
2	cubemap
3	mesh

Table 29

10.4.1.31.20.2. ProjectionPrivate Element

name: ProjectionPrivate

path: "\Segment\Tracks\TrackEntry\Video\Projection\ProjectionPrivate"

id: 0x7672

maxOccurs: 1

type: binary

minver: 4

definition: Private data that only applies to a specific projection.

* If "ProjectionType" equals 0 (Rectangular), then this element must not be present.

* If "ProjectionType" equals 1 (Equirectangular), then this element must be present and contain the same binary data that would be stored inside an ISOBMFF Equirectangular Projection Box ('equi').

* If "ProjectionType" equals 2 (Cubemap), then this element must be present and contain the same binary data that would be stored inside an ISOBMFF Cubemap Projection Box ('cbmp').

* If "ProjectionType" equals 3 (Mesh), then this element must be present and contain the same binary data that would be stored inside an ISOBMFF Mesh Projection Box ('mshp').

usage notes: ISOBMFF box size and fourcc fields are not included in the binary data, but the FullBox version and flag fields are. This is to avoid redundant framing information while preserving versioning and semantics between the two container formats.

10.4.1.31.20.3. ProjectionPoseYaw Element

name: ProjectionPoseYaw

path: "\Segment\Tracks\TrackEntry\Video\Projection\ProjectionPoseYaw"

id: 0x7673

minOccurs: 1

maxOccurs: 1

default: 0x0p+0

type: float

minver: 4

definition: Specifies a yaw rotation to the projection.

Value represents a clockwise rotation, in degrees, around the up vector. This rotation must be applied before any "ProjectionPosePitch" or "ProjectionPoseRoll" rotations. The value of this field should be in the -180 to 180 degree range.

10.4.1.31.20.4. ProjectionPosePitch Element

name: ProjectionPosePitch

path: "\Segment\Tracks\TrackEntry\Video\Projection\ProjectionPosePitch"

id: 0x7674

minOccurs: 1

maxOccurs: 1

default: 0x0p+0

type: float

minver: 4

definition: Specifies a pitch rotation to the projection.

Value represents a counter-clockwise rotation, in degrees, around the right vector. This rotation must be applied after the "ProjectionPoseYaw" rotation and before the "ProjectionPoseRoll" rotation. The value of this field should be in the -90 to 90 degree range.

10.4.1.31.20.5. ProjectionPoseRoll Element

name: ProjectionPoseRoll

path: "\Segment\Tracks\TrackEntry\Video\Projection\ProjectionPoseRoll"

id: 0x7675

minOccurs: 1

maxOccurs: 1

default: 0x0p+0

type: float

minver: 4

definition: Specifies a roll rotation to the projection.

Value represents a counter-clockwise rotation, in degrees, around the forward vector. This rotation must be applied after the "ProjectionPoseYaw" and "ProjectionPosePitch" rotations. The value of this field should be in the -180 to 180 degree range.

10.4.1.32. Audio Element

name: Audio

path: "\Segment\Tracks\TrackEntry\Audio"

id: 0xE1

maxOccurs: 1

type: master

definition: Audio settings.

10.4.1.32.1. SamplingFrequency Element

name: SamplingFrequency

path: "\Segment\Tracks\TrackEntry\Audio\SamplingFrequency"

id: 0xB5

minOccurs: 1

maxOccurs: 1

range: > 0x0p+0

default: 0x1.f4p+12

type: float

definition: Sampling frequency in Hz.

10.4.1.32.2. OutputSamplingFrequency Element

name: OutputSamplingFrequency

path: "\Segment\Tracks\TrackEntry\Audio\OutputSamplingFrequency"

id: 0x78B5

maxOccurs: 1

range: > 0x0p+0

default: see implementation notes

type: float

definition: Real output sampling frequency in Hz (used for SBR techniques).

implementation notes:

attribute	note
default	The default value for OutputSamplingFrequency of the same TrackEntry is equal to the SamplingFrequency.

Table 30

10.4.1.32.3. Channels Element

name: Channels

path: "\Segment\Tracks\TrackEntry\Audio\Channels"

id: 0x9F

minOccurs: 1

maxOccurs: 1

range: not 0

default: 1

type: uinteger

definition: Numbers of channels in the track.

10.4.1.32.4. ChannelPositions Element

name: ChannelPositions

path: "\Segment\Tracks\TrackEntry\Audio\ChannelPositions"

id: 0x7D7B

maxOccurs: 1

type: binary

minver: 0

maxver: 0

definition: Table of horizontal angles for each successive channel.

10.4.1.32.5. BitDepth Element

name: BitDepth

path: "\Segment\Tracks\TrackEntry\Audio\BitDepth"

id: 0x6264

maxOccurs: 1

range: not 0

type: uinteger

definition: Bits per sample, mostly used for PCM.

10.4.1.33. TrackOperation Element

name: TrackOperation

path: "\Segment\Tracks\TrackEntry\TrackOperation"

id: 0xE2

maxOccurs: 1

type: master

minver: 3

definition: Operation that needs to be applied on tracks to create this virtual track. For more details look at Section 26.3.

10.4.1.33.1. TrackCombinePlanes Element

name: TrackCombinePlanes

path: "\Segment\Tracks\TrackEntry\TrackOperation\TrackCombinePlanes"

id: 0xE3

maxOccurs: 1

type: master

minver: 3

definition: Contains the list of all video plane tracks that need to be combined to create this 3D track

10.4.1.33.1.1. TrackPlane Element

name: TrackPlane

path: "\Segment\Tracks\TrackEntry\TrackOperation\TrackCombinePlanes\TrackPlane"

id: 0xE4

minOccurs: 1

type: master

minver: 3

definition: Contains a video plane track that need to be combined to create this 3D track

10.4.1.33.1.2. TrackPlaneUID Element

name: TrackPlaneUID

path: "\Segment\Tracks\TrackEntry\TrackOperation\TrackCombinePlanes\
TrackPlane\TrackPlaneUID"

id: 0xE5

minOccurs: 1

maxOccurs: 1

range: not 0

type: uinteger

minver: 3

definition: The trackUID number of the track representing the plane.

10.4.1.33.1.3. TrackPlaneType Element

name: TrackPlaneType

path: "\Segment\Tracks\TrackEntry\TrackOperation\TrackCombinePlanes\
TrackPlane\TrackPlaneType"

id: 0xE6

minOccurs: 1

maxOccurs: 1

type: uinteger

minver: 3

definition: The kind of plane this track corresponds to.

restrictions:

value	label
0	left eye
1	right eye
2	background

Table 31

10.4.1.33.2. TrackJoinBlocks Element

name: TrackJoinBlocks

path: "\Segment\Tracks\TrackEntry\TrackOperation\TrackJoinBlocks"

id: 0xE9

maxOccurs: 1

type: master

minver: 3

definition: Contains the list of all tracks whose Blocks need to be combined to create this virtual track

10.4.1.33.2.1. TrackJoinUID Element

name: TrackJoinUID

path: "\Segment\Tracks\TrackEntry\TrackOperation\TrackJoinBlocks\TrackJoinUID"

id: 0xED

minOccurs: 1

range: not 0

type: uinteger

minver: 3

definition: The trackUID number of a track whose blocks are used to create this virtual track.

10.4.1.34. TrickTrackUID Element

name: TrickTrackUID

path: "\Segment\Tracks\TrackEntry\TrickTrackUID"

id: 0xC0

maxOccurs: 1

type: uinteger

minver: 0

maxver: 0

definition: DivX trick track extensions

10.4.1.35. TrickTrackSegmentUID Element

name: TrickTrackSegmentUID

path: "\Segment\Tracks\TrackEntry\TrickTrackSegmentUID"

id: 0xC1

maxOccurs: 1

type: binary

minver: 0

maxver: 0

definition: DivX trick track extensions

10.4.1.36. TrickTrackFlag Element

name: TrickTrackFlag

path: "\Segment\Tracks\TrackEntry\TrickTrackFlag"

id: 0xC6

maxOccurs: 1

default: 0

type: uinteger

minver: 0

maxver: 0

definition: DivX trick track extensions

10.4.1.37. TrickMasterTrackUID Element

name: TrickMasterTrackUID

path: "\Segment\Tracks\TrackEntry\TrickMasterTrackUID"

id: 0xC7

maxOccurs: 1

type: uinteger

minver: 0

maxver: 0

definition: DivX trick track extensions

10.4.1.38. TrickMasterTrackSegmentUID Element

name: TrickMasterTrackSegmentUID

path: "\Segment\Tracks\TrackEntry\TrickMasterTrackSegmentUID"

id: 0xC4

maxOccurs: 1

type: binary

minver: 0

maxver: 0

definition: DivX trick track extensions

10.4.1.39. ContentEncodings Element

name: ContentEncodings

path: "\Segment\Tracks\TrackEntry\ContentEncodings"

id: 0x6D80

maxOccurs: 1

type: master

definition: Settings for several content encoding mechanisms like compression or encryption.

10.4.1.39.1. ContentEncoding Element

name: ContentEncoding

path: "\Segment\Tracks\TrackEntry\ContentEncodings\ContentEncoding"

id: 0x6240

minOccurs: 1

type: master

definition: Settings for one content encoding like compression or encryption.

10.4.1.39.1.1. ContentEncodingOrder Element

name: ContentEncodingOrder

path: "\Segment\Tracks\TrackEntry\ContentEncodings\ContentEncoding\ContentEncodingOrder"

id: 0x5031

minOccurs: 1

maxOccurs: 1

default: 0

type: uinteger

definition: Tells when this modification was used during encoding/

muxing starting with 0 and counting upwards. The decoder/demuxer has to start with the highest order number it finds and work its way down. This value has to be unique over all ContentEncodingOrder Elements in the TrackEntry that contains this ContentEncodingOrder element.

10.4.1.39.1.2. ContentEncodingScope Element

name: ContentEncodingScope

path: "\Segment\Tracks\TrackEntry\ContentEncodings\ContentEncoding\ContentEncodingScope"

id: 0x5032

minOccurs: 1

maxOccurs: 1

range: not 0

default: 1

type: uinteger

definition: A bit field that describes which Elements have been modified in this way. Values (big-endian) can be OR'ed.

restrictions:

value	label
1	All frame contents, excluding lacing data
2	The track's private data
4	The next ContentEncoding (next "ContentEncodingOrder". Either the data inside "ContentCompression" and/or "ContentEncryption")

Table 32

10.4.1.39.1.3. ContentEncodingType Element

name: ContentEncodingType

path: "\Segment\Tracks\TrackEntry\ContentEncodings\ContentEncoding\ContentEncodingType"

id: 0x5033

minOccurs: 1

maxOccurs: 1

default: 0

type: uinteger

definition: A value describing what kind of transformation is applied.

restrictions:

value	label
0	Compression
1	Encryption

Table 33

10.4.1.39.1.4. ContentCompression Element

name: ContentCompression

path: "\Segment\Tracks\TrackEntry\ContentEncodings\ContentEncoding\ContentCompression"

id: 0x5034

maxOccurs: 1

type: master

definition: Settings describing the compression used. This Element MUST be present if the value of ContentEncodingType is 0 and absent otherwise. Each block MUST be decompressable even if no previous block is available in order not to prevent seeking.

10.4.1.39.1.5. ContentCompAlgo Element

name: ContentCompAlgo

path: "\Segment\Tracks\TrackEntry\ContentEncodings\ContentEncoding\ContentCompression\ContentCompAlgo"

id: 0x4254

minOccurs: 1

maxOccurs: 1

default: 0

type: uinteger

definition: The compression algorithm used.

restrictions:

value	label
0	zlib
1	bzlib
2	lzo1x
3	Header Stripping

Table 34

10.4.1.39.1.6. ContentCompSettings Element

name: ContentCompSettings

path: "\Segment\Tracks\TrackEntry\ContentEncodings\ContentEncoding\ContentCompression\ContentCompSettings"

id: 0x4255

maxOccurs: 1

type: binary

definition: Settings that might be needed by the decompressor. For Header Stripping ("ContentCompAlgo"=3), the bytes that were removed from the beginning of each frames of the track.

10.4.1.39.1.7. ContentEncryption Element

name: ContentEncryption

path: "\Segment\Tracks\TrackEntry\ContentEncodings\ContentEncoding\ContentEncryption"

id: 0x5035

maxOccurs: 1

type: master

definition: Settings describing the encryption used. This Element MUST be present if the value of "ContentEncodingType" is 1 (encryption) and MUST be ignored otherwise.

10.4.1.39.1.8. ContentEncAlgo Element

name: ContentEncAlgo

path: "\Segment\Tracks\TrackEntry\ContentEncodings\ContentEncoding\ContentEncryption\ContentEncAlgo"

id: 0x47E1

minOccurs: 1

maxOccurs: 1

default: 0

type: uinteger

definition: The encryption algorithm used. The value "0" means that the contents have not been encrypted but only signed.

restrictions:

value	label
0	Not encrypted
1	DES - FIPS 46-3
2	Triple DES - RFC 1851
3	Twofish
4	Blowfish
5	AES - FIPS 187

Table 35

10.4.1.39.1.9. ContentEncKeyID Element

name: ContentEncKeyID

path: "\Segment\Tracks\TrackEntry\ContentEncodings\ContentEncoding\ContentEncryption\ContentEncKeyID"

id: 0x47E2

maxOccurs: 1

type: binary

definition: For public key algorithms this is the ID of the public key the the data was encrypted with.

10.4.1.39.1.10. ContentEncAESSettings Element

name: ContentEncAESSettings

path: "\Segment\Tracks\TrackEntry\ContentEncodings\ContentEncoding\ContentEncryption\ContentEncAESSettings"

id: 0x47E7

maxOccurs: 1

type: master

minver: 4

definition: Settings describing the encryption algorithm used. If "ContentEncAlgo" != 5 this MUST be ignored.

10.4.1.39.1.11. AESSettingsCipherMode Element

name: AESSettingsCipherMode

path: "\Segment\Tracks\TrackEntry\ContentEncodings\ContentEncoding\ContentEncryption\ContentEncAESSettings\AESSettingsCipherMode"

id: 0x47E8

minOccurs: 1

maxOccurs: 1

type: uinteger

minver: 4

definition: The AES cipher mode used in the encryption.

restrictions:

value	label
1	AES-CTR / Counter, NIST SP 800-38A
2	AES-CBC / Cipher Block Chaining, NIST SP 800-38A

Table 36

10.4.1.39.1.12. ContentSignature Element

name: ContentSignature

path: "\Segment\Tracks\TrackEntry\ContentEncodings\ContentEncoding\ContentEncryption\ContentSignature"

id: 0x47E3

maxOccurs: 1

type: binary

definition: A cryptographic signature of the contents.

10.4.1.39.1.13. ContentSigKeyID Element

name: ContentSigKeyID

path: "\Segment\Tracks\TrackEntry\ContentEncodings\ContentEncoding\ContentEncryption\ContentSigKeyID"

id: 0x47E4

maxOccurs: 1

type: binary

definition: This is the ID of the private key the data was signed with.

10.4.1.39.1.14. ContentSigAlgo Element

name: ContentSigAlgo

path: "\Segment\Tracks\TrackEntry\ContentEncodings\ContentEncoding\ContentEncryption\ContentSigAlgo"

id: 0x47E5

maxOccurs: 1

default: 0

type: uinteger

definition: The algorithm used for the signature.

restrictions:

value	label
0	Not signed
1	RSA

Table 37

10.4.1.39.1.15. ContentSigHashAlgo Element

name: ContentSigHashAlgo

path: "\Segment\Tracks\TrackEntry\ContentEncodings\ContentEncoding\ContentEncryption\ContentSigHashAlgo"

id: 0x47E6

maxOccurs: 1

default: 0

type: uinteger

definition: The hash algorithm used for the signature.

restrictions:

value	label
0	Not signed
1	SHA1-160
2	MD5

Table 38

10.5. Cues Element

name: Cues

path: "\Segment\Cues"

id: 0x1C53BB6B

minOccurs: see implementation notes

maxOccurs: 1

type: master

definition: A Top-Level Element to speed seeking access. All entries are local to the Segment.

implementation notes:

attribute	note
minOccurs	This Element SHOULD be set when the Segment is not transmitted as a live stream (see #livestreaming).

Table 39

10.5.1. CuePoint Element

name: CuePoint

path: "\Segment\Cues\CuePoint"

id: 0xBB

minOccurs: 1

type: master

definition: Contains all information relative to a seek point in the Segment.

10.5.1.1. CueTime Element

name: CueTime

path: "\Segment\Cues\CuePoint\CueTime"

id: 0xB3

minOccurs: 1

maxOccurs: 1

type: uinteger

definition: Absolute timestamp according to the Segment time base.

10.5.1.2. CueTrackPositions Element

name: CueTrackPositions

path: "\Segment\Cues\CuePoint\CueTrackPositions"

id: 0xB7

minOccurs: 1

type: master

definition: Contain positions for different tracks corresponding to the timestamp.

10.5.1.2.1. CueTrack Element

name: CueTrack

path: "\Segment\Cues\CuePoint\CueTrackPositions\CueTrack"

id: 0xF7

minOccurs: 1

maxOccurs: 1

range: not 0

type: uinteger

definition: The track for which a position is given.

10.5.1.2.2. CueClusterPosition Element

name: CueClusterPosition

path: "\Segment\Cues\CuePoint\CueTrackPositions\CueClusterPosition"

id: 0xF1

minOccurs: 1

maxOccurs: 1

type: uinteger

definition: The Segment Position of the Cluster containing the associated Block.

10.5.1.2.3. CueRelativePosition Element

name: CueRelativePosition

path: "\Segment\Cues\CuePoint\CueTrackPositions\CueRelativePosition"

id: 0xF0

maxOccurs: 1

type: uinteger

minver: 4

definition: The relative position inside the Cluster of the referenced SimpleBlock or BlockGroup with 0 being the first possible position for an Element inside that Cluster.

10.5.1.2.4. CueDuration Element

name: CueDuration

path: "\Segment\Cues\CuePoint\CueTrackPositions\CueDuration"

id: 0xB2

maxOccurs: 1

type: uinteger

minver: 4

definition: The duration of the block according to the Segment time base. If missing the track's DefaultDuration does not apply and no duration information is available in terms of the cues.

10.5.1.2.5. CueBlockNumber Element

name: CueBlockNumber

path: "\Segment\Cues\CuePoint\CueTrackPositions\CueBlockNumber"

id: 0x5378

maxOccurs: 1

range: not 0

default: 1

type: uinteger

definition: Number of the Block in the specified Cluster.

10.5.1.2.6. CueCodecState Element

name: CueCodecState

path: "\Segment\Cues\CuePoint\CueTrackPositions\CueCodecState"

id: 0xEA

maxOccurs: 1

default: 0

type: uinteger

minver: 2

definition: The Segment Position of the Codec State corresponding to this Cue Element. 0 means that the data is taken from the initial Track Entry.

10.5.1.2.7. CueReference Element

name: CueReference

path: "\Segment\Cues\CuePoint\CueTrackPositions\CueReference"

id: 0xDB

type: master

minver: 2

definition: The Clusters containing the referenced Blocks.

10.5.1.2.7.1. CueRefTime Element

name: CueRefTime

path: "\Segment\Cues\CuePoint\CueTrackPositions\CueReference\CueRefTime"

id: 0x96

minOccurs: 1

maxOccurs: 1

type: uinteger

minver: 2

definition: Timestamp of the referenced Block.

10.5.1.2.7.2. CueRefCluster Element

name: CueRefCluster

path: "\Segment\Cues\CuePoint\CueTrackPositions\CueReference\CueRefCluster"

id: 0x97

minOccurs: 1

maxOccurs: 1

type: uinteger

minver: 0

maxver: 0

definition: The Segment Position of the Cluster containing the referenced Block.

10.5.1.2.7.3. CueRefNumber Element

name: CueRefNumber

path: "\Segment\Cues\CuePoint\CueTrackPositions\CueReference\CueRefNumber"

id: 0x535F

maxOccurs: 1

range: not 0

default: 1

type: uinteger

minver: 0

maxver: 0

definition: Number of the referenced Block of Track X in the specified Cluster.

10.5.1.2.7.4. CueRefCodecState Element

name: CueRefCodecState

path: "\Segment\Cues\CuePoint\CueTrackPositions\CueReference\CueRefCodecState"

id: 0xEB

maxOccurs: 1

default: 0

type: uinteger

minver: 0

maxver: 0

definition: The Segment Position of the Codec State corresponding to this referenced Element. 0 means that the data is taken from the initial Track Entry.

10.6. Attachments Element

name: Attachments

path: "\Segment\Attachments"

id: 0x1941A469

maxOccurs: 1

type: master

definition: Contain attached files.

10.6.1. AttachedFile Element

name: AttachedFile

path: "\Segment\Attachments\AttachedFile"

id: 0x61A7

minOccurs: 1

type: master

definition: An attached file.

10.6.1.1. FileDescription Element

name: FileDescription

path: "\Segment\Attachments\AttachedFile\FileDescription"

id: 0x467E

maxOccurs: 1

type: utf-8

definition: A human-friendly name for the attached file.

10.6.1.2. FileName Element

name: FileName

path: "\Segment\Attachments\AttachedFile\FileName"

id: 0x466E

minOccurs: 1

maxOccurs: 1

type: utf-8

definition: Filename of the attached file.

10.6.1.3. FileMimeType Element

name: FileMimeType

path: "\Segment\Attachments\AttachedFile\FileMimeType"

id: 0x4660

minOccurs: 1

maxOccurs: 1

type: string

definition: MIME type of the file.

10.6.1.4. FileData Element

name: FileData

path: "\Segment\Attachments\AttachedFile\FileData"

id: 0x465C

minOccurs: 1

maxOccurs: 1

type: binary

definition: The data of the file.

10.6.1.5. FileUID Element

name: FileUID

path: "\Segment\Attachments\AttachedFile\FileUID"

id: 0x46AE

minOccurs: 1

maxOccurs: 1

range: not 0

type: uinteger

definition: Unique ID representing the file, as random as possible.

10.6.1.6. FileReferral Element

name: FileReferral

path: "\Segment\Attachments\AttachedFile\FileReferral"

id: 0x4675

maxOccurs: 1

type: binary

minver: 0

maxver: 0

definition: A binary value that a track/codec can refer to when the attachment is needed.

10.6.1.7. FileUsedStartTime Element

name: FileUsedStartTime

path: "\Segment\Attachments\AttachedFile\FileUsedStartTime"

id: 0x4661

maxOccurs: 1

type: uinteger

minver: 0

maxver: 0

definition: DivX font extension

10.6.1.8. FileUsedEndTime Element

name: FileUsedEndTime

path: "\Segment\Attachments\AttachedFile\FileUsedEndTime"

id: 0x4662

maxOccurs: 1

type: uinteger

minver: 0

maxver: 0

definition: DivX font extension

10.7. Chapters Element

name: Chapters

path: "\Segment\Chapters"

id: 0x1043A770

maxOccurs: 1

type: master

recurring: 1

definition: A system to define basic menus and partition data. For more detailed information, look at the Chapters explanation in Section 12.

10.7.1. EditionEntry Element

name: EditionEntry

path: "\Segment\Chapters\EditionEntry"

id: 0x45B9

minOccurs: 1

type: master

definition: Contains all information about a Segment edition.

10.7.1.1. EditionUID Element

name: EditionUID

path: "\Segment\Chapters\EditionEntry\EditionUID"

id: 0x45BC

maxOccurs: 1

range: not 0

type: uinteger

definition: A unique ID to identify the edition. It’s useful for tagging an edition.

10.7.1.2. EditionFlagHidden Element

name: EditionFlagHidden

path: "\Segment\Chapters\EditionEntry\EditionFlagHidden"

id: 0x45BD

minOccurs: 1

maxOccurs: 1

range: 0-1

default: 0

type: uinteger

definition: If an edition is hidden (1), it SHOULD NOT be available to the user interface (but still to Control Tracks; see Section 12.1.1 on Chapter flags). (1 bit)

10.7.1.3. EditionFlagDefault Element

name: EditionFlagDefault

path: "\Segment\Chapters\EditionEntry\EditionFlagDefault"

id: 0x45DB

minOccurs: 1

maxOccurs: 1

range: 0-1

default: 0

type: uinteger

definition: If a flag is set (1) the edition SHOULD be used as the default one. (1 bit)

10.7.1.4. EditionFlagOrdered Element

name: EditionFlagOrdered

path: "\Segment\Chapters\EditionEntry\EditionFlagOrdered"

id: 0x45DD

maxOccurs: 1

range: 0-1

default: 0

type: uinteger

definition: Specify if the chapters can be defined multiple times and the order to play them is enforced. (1 bit)

10.7.1.5. ChapterAtom Element

name: ChapterAtom

path: "\Segment\Chapters\EditionEntry\+ChapterAtom"

id: 0xB6

minOccurs: 1

type: master

recursive: 1

definition: Contains the atom information to use as the chapter atom (apply to all tracks).

10.7.1.5.1. ChapterUID Element

name: ChapterUID

path: "\Segment\Chapters\EditionEntry\+ChapterAtom\ChapterUID"

id: 0x73C4

minOccurs: 1

maxOccurs: 1

range: not 0

type: uinteger

definition: A unique ID to identify the Chapter.

10.7.1.5.2. ChapterStringUID Element

name: ChapterStringUID

path: "\Segment\Chapters\EditionEntry\+ChapterAtom\ChapterStringUID"

id: 0x5654

maxOccurs: 1

type: utf-8

minver: 3

definition: A unique string ID to identify the Chapter. Use for WebVTT cue identifier storage.

10.7.1.5.3. ChapterTimeStart Element

name: ChapterTimeStart

path: "\Segment\Chapters\EditionEntry\+ChapterAtom\ChapterTimeStart"

id: 0x91

minOccurs: 1

maxOccurs: 1

type: uinteger

definition: Timestamp of the start of Chapter (not scaled).

10.7.1.5.4. ChapterTimeEnd Element

name: ChapterTimeEnd

path: "\Segment\Chapters\EditionEntry\+ChapterAtom\ChapterTimeEnd"

id: 0x92

maxOccurs: 1

type: uinteger

definition: Timestamp of the end of Chapter (timestamp excluded, not scaled).

10.7.1.5.5. ChapterFlagHidden Element

name: ChapterFlagHidden

path: "\Segment\Chapters\EditionEntry\+ChapterAtom\ChapterFlagHidden"

id: 0x98

minOccurs: 1

maxOccurs: 1

range: 0-1

default: 0

type: uinteger

definition: If a chapter is hidden (1), it SHOULD NOT be available to the user interface (but still to Control Tracks; see Section 12.1.1 on Chapter flags). (1 bit)

10.7.1.5.6. ChapterFlagEnabled Element

name: ChapterFlagEnabled

path: "\Segment\Chapters\EditionEntry\+ChapterAtom\ChapterFlagEnabled"

id: 0x4598

minOccurs: 1

maxOccurs: 1

range: 0-1

default: 1

type: uinteger

definition: Specify whether the chapter is enabled. It can be enabled/disabled by a Control Track. When disabled, the movie SHOULD skip all the content between the TimeStart and TimeEnd of this chapter; see Section 12.1.1 on Chapter flags. (1 bit)

10.7.1.5.7. ChapterSegmentUID Element

name: ChapterSegmentUID

path: "\Segment\Chapters\EditionEntry\+ChapterAtom\ChapterSegmentUID"

id: 0x6E67

minOccurs: see implementation notes

maxOccurs: 1

range: >0

type: binary

definition: The SegmentUID of another Segment to play during this chapter.

implementation notes:

attribute	note
minOccurs	ChapterSegmentUID MUST be set (minOccurs=1) if ChapterSegmentEditionUID is used.

Table 40

10.7.1.5.8. ChapterSegmentEditionUID Element

name: ChapterSegmentEditionUID

path: "\Segment\Chapters\EditionEntry\+ChapterAtom\ChapterSegmentEditionUID"

id: 0x6EBC

maxOccurs: 1

range: not 0

type: uinteger

definition: The EditionUID to play from the Segment linked in

ChapterSegmentUID. If ChapterSegmentEditionUID is undeclared, then no Edition of the linked Segment is used.

10.7.1.5.9. ChapterPhysicalEquiv Element

name: ChapterPhysicalEquiv

path: "\Segment\Chapters\EditionEntry\+ChapterAtom\ChapterPhysicalEquiv"

id: 0x63C3

maxOccurs: 1

type: uinteger

definition: Specify the physical equivalent of this ChapterAtom like "DVD" (60) or "SIDE" (50); see Section 6.2.2 for a complete list of values.

10.7.1.5.10. ChapterTrack Element

name: ChapterTrack

path: "\Segment\Chapters\EditionEntry\+ChapterAtom\ChapterTrack"

id: 0x8F

maxOccurs: 1

type: master

definition: List of tracks on which the chapter applies. If this Element is not present, all tracks apply

10.7.1.5.10.1. ChapterTrackUID Element

name: ChapterTrackUID

path: "\Segment\Chapters\EditionEntry\+ChapterAtom\ChapterTrack\ChapterTrackUID"

id: 0x89

minOccurs: 1

range: not 0

type: uinteger

definition: UID of the Track to apply this chapter to. In the absence of a control track, choosing this chapter will select the listed Tracks and deselect unlisted tracks. Absence of this Element indicates that the Chapter SHOULD be applied to any currently used Tracks.

10.7.1.5.11. ChapterDisplay Element

name: ChapterDisplay

path: "\Segment\Chapters\EditionEntry\+ChapterAtom\ChapterDisplay"

id: 0x80

type: master

definition: Contains all possible strings to use for the chapter display.

10.7.1.5.11.1. ChapString Element

name: ChapString

path: "\Segment\Chapters\EditionEntry\+ChapterAtom\ChapterDisplay\ChapString"

id: 0x85

minOccurs: 1

maxOccurs: 1

type: utf-8

definition: Contains the string to use as the chapter atom.

10.7.1.5.11.2. ChapLanguage Element

name: ChapLanguage

path: "\Segment\Chapters\EditionEntry\+ChapterAtom\ChapterDisplay\ChapLanguage"

id: 0x437C

minOccurs: 1

default: eng

type: string

definition: The languages corresponding to the string, in the bibliographic ISO-639-2 form. This Element MUST be ignored if the ChapLanguageIETF Element is used within the same ChapterDisplay Element.

10.7.1.5.11.3. ChapLanguageIETF Element

name: ChapLanguageIETF

path: "\Segment\Chapters\EditionEntry\+ChapterAtom\ChapterDisplay\ChapLanguageIETF"

id: 0x437D

type: string

minver: 4

definition: Specifies the language used in the ChapString according to BCP 47 and using the IANA Language Subtag Registry. If this Element is used, then any ChapLanguage Elements used in the same ChapterDisplay MUST be ignored.

10.7.1.5.11.4. ChapCountry Element

name: ChapCountry

path: "\Segment\Chapters\EditionEntry\+ChapterAtom\ChapterDisplay\ChapCountry"

id: 0x437E

type: string

definition: The countries corresponding to the string, same 2 octets as in Internet domains. This Element MUST be ignored if the ChapLanguageIETF Element is used within the same ChapterDisplay Element.

10.7.1.5.12. ChapProcess Element

name: ChapProcess

path: "\Segment\Chapters\EditionEntry\+ChapterAtom\ChapProcess"

id: 0x6944

type: master

definition: Contains all the commands associated to the Atom.

10.7.1.5.12.1. ChapProcessCodecID Element

name: ChapProcessCodecID

path: "\Segment\Chapters\EditionEntry\+ChapterAtom\ChapProcess\ChapProcessCodecID"

id: 0x6955

minOccurs: 1

maxOccurs: 1

default: 0

type: uinteger

definition: Contains the type of the codec used for the processing. A value of 0 means native Matroska processing (to be defined), a value of 1 means the DVD command set is used; see Section 12.2.2 on DVD menus. More codec IDs can be added later.

10.7.1.5.12.2. ChapProcessPrivate Element

name: ChapProcessPrivate

path: "\Segment\Chapters\EditionEntry\+ChapterAtom\ChapProcess\ChapProcessPrivate"

id: 0x450D

maxOccurs: 1

type: binary

definition: Some optional data attached to the ChapProcessCodecID information. For ChapProcessCodecID = 1, it is the "DVD level" equivalent; see Section 12.2.2 on DVD menus.

10.7.1.5.12.3. ChapProcessCommand Element

name: ChapProcessCommand

path: "\Segment\Chapters\EditionEntry\+ChapterAtom\ChapProcess\ChapProcessCommand"

id: 0x6911

type: master

definition: Contains all the commands associated to the Atom.

10.7.1.5.12.4. ChapProcessTime Element

name: ChapProcessTime

path: "\Segment\Chapters\EditionEntry\+ChapterAtom\ChapProcess\ChapProcessCommand\ChapProcessTime"

id: 0x6922

minOccurs: 1

maxOccurs: 1

type: uinteger

definition: Defines when the process command SHOULD be handled

restrictions:

value	label
0	during the whole chapter
1	before starting playback
2	after playback of the chapter

Table 41

10.7.1.5.12.5. ChapProcessData Element

name: ChapProcessData

path: "\Segment\Chapters\EditionEntry\+ChapterAtom\ChapProcess\ChapProcessCommand\ChapProcessData"

id: 0x6933

minOccurs: 1

maxOccurs: 1

type: binary

definition: Contains the command information. The data SHOULD be interpreted depending on the ChapProcessCodecID value. For ChapProcessCodecID = 1, the data correspond to the binary DVD cell pre/post commands; see Section 12.2.2 on DVD menus.

10.8. Tags Element

name: Tags

path: "\Segment\Tags"

id: 0x1254C367

type: master

definition: Element containing metadata describing Tracks, Editions, Chapters, Attachments, or the Segment as a whole. A list of valid tags can be found in [I-D.ietf-cellar-tags].

10.8.1. Tag Element

name: Tag

path: "\Segment\Tags\Tag"

id: 0x7373

minOccurs: 1

type: master

definition: A single metadata descriptor.

10.8.1.1. Targets Element

name: Targets

path: "\Segment\Tags\Tag\Targets"

id: 0x63C0

minOccurs: 1

maxOccurs: 1

type: master

definition: Specifies which other elements the metadata represented by the Tag applies to. If empty or not present, then the Tag describes everything in the Segment.

10.8.1.1.1. TargetTypeValue Element

name: TargetTypeValue

path: "\Segment\Tags\Tag\Targets\TargetTypeValue"

id: 0x68CA

maxOccurs: 1

default: 50

type: uinteger

definition: A number to indicate the logical level of the target.

restrictions:

value	label	documentation
70	COLLECTION	The highest hierarchical level that tags can describe.
60	EDITION / ISSUE / VOLUME / OPUS / SEASON / SEQUEL	A list of lower levels grouped together.
50	ALBUM / OPERA / CONCERT / MOVIE / EPISODE / CONCERT	The most common grouping level of music and video (equals to an episode for TV series).
40	PART / SESSION	When an album or episode has different logical parts.
30	TRACK / SONG / CHAPTER	The common parts of an album or movie.
20	SUBTRACK / PART / MOVEMENT / SCENE	Corresponds to parts of a track for audio (like a movement).
10	SHOT	The lowest hierarchy found in music or movies.

Table 42

10.8.1.1.2. TargetType Element

name: TargetType

path: "\Segment\Tags\Tag\Targets\TargetType"

id: 0x63CA

maxOccurs: 1

type: string

definition: An informational string that can be used to display the logical level of the target like "ALBUM", "TRACK", "MOVIE", "CHAPTER", etc ; see Section 6.4 of [I-D.ietf-cellar-tags].

restrictions:

value	label
COLLECTION	COLLECTION
EDITION	EDITION
ISSUE	ISSUE
VOLUME	VOLUME
OPUS	OPUS
SEASON	SEASON
SEQUEL	SEQUEL
ALBUM	ALBUM
OPERA	OPERA
CONCERT	CONCERT
MOVIE	MOVIE
EPISODE	EPISODE
PART	PART
SESSION	SESSION
TRACK	TRACK
SONG	SONG
CHAPTER	CHAPTER
SUBTRACK	SUBTRACK
PART	PART
MOVEMENT	MOVEMENT
SCENE	SCENE
SHOT	SHOT

Table 43

10.8.1.1.3. TagTrackUID Element

name: TagTrackUID

path: "\Segment\Tags\Tag\Targets\TagTrackUID"

id: 0x63C5

default: 0

type: uinteger

definition: A unique ID to identify the Track(s) the tags belong to. If the value is 0 at this level, the tags apply to all tracks in the Segment.

10.8.1.1.4. TagEditionUID Element

name: TagEditionUID

path: "\Segment\Tags\Tag\Targets\TagEditionUID"

id: 0x63C9

default: 0

type: uinteger

definition: A unique ID to identify the EditionEntry(s) the tags belong to. If the value is 0 at this level, the tags apply to all editions in the Segment.

10.8.1.1.5. TagChapterUID Element

name: TagChapterUID

path: "\Segment\Tags\Tag\Targets\TagChapterUID"

id: 0x63C4

default: 0

type: uinteger

definition: A unique ID to identify the Chapter(s) the tags belong

to. If the value is 0 at this level, the tags apply to all chapters in the Segment.

10.8.1.1.6. TagAttachmentUID Element

name: TagAttachmentUID

path: "\Segment\Tags\Tag\Targets\TagAttachmentUID"

id: 0x63C6

default: 0

type: uinteger

definition: A unique ID to identify the Attachment(s) the tags belong to. If the value is 0 at this level, the tags apply to all the attachments in the Segment.

10.8.1.2. SimpleTag Element

name: SimpleTag

path: "\Segment\Tags\Tag\+SimpleTag"

id: 0x67C8

minOccurs: 1

type: master

recursive: 1

definition: Contains general information about the target.

10.8.1.2.1. TagName Element

name: TagName

path: "\Segment\Tags\Tag\+SimpleTag\TagName"

id: 0x45A3

minOccurs: 1

maxOccurs: 1

type: utf-8

definition: The name of the Tag that is going to be stored.

10.8.1.2.2. TagLanguage Element

name: TagLanguage

path: "\Segment\Tags\Tag\+SimpleTag\TagLanguage"

id: 0x447A

minOccurs: 1

maxOccurs: 1

default: und

type: string

definition: Specifies the language of the tag specified, in the Matroska languages form; see Section 6.2.1 on language codes. This Element MUST be ignored if the TagLanguageIETF Element is used within the same SimpleTag Element.

10.8.1.2.3. TagLanguageIETF Element

name: TagLanguageIETF

path: "\Segment\Tags\Tag\+SimpleTag\TagLanguageIETF"

id: 0x447B

maxOccurs: 1

type: string

minver: 4

definition: Specifies the language used in the TagString according to BCP 47 and using the IANA Language Subtag Registry. If this Element is used, then any TagLanguage Elements used in the same SimpleTag MUST be ignored.

10.8.1.2.4. TagDefault Element

name: TagDefault

path: "\Segment\Tags\Tag\+SimpleTag\TagDefault"

id: 0x4484

minOccurs: 1

maxOccurs: 1

range: 0-1

default: 1

type: uinteger

definition: A boolean value to indicate if this is the default/ original language to use for the given tag.

10.8.1.2.5. TagString Element

name: TagString

path: "\Segment\Tags\Tag\+SimpleTag\TagString"

id: 0x4487

maxOccurs: 1

type: utf-8

definition: The value of the Tag.

10.8.1.2.6. TagBinary Element

name: TagBinary

path: "\Segment\Tags\Tag\+SimpleTag\TagBinary"

id: 0x4485

maxOccurs: 1

type: binary

definition: The values of the Tag, if it is binary. Note that this cannot be used in the same SimpleTag as TagString.

11. Matroska Element Ordering

Except for the "EBML Header" and the "CRC-32 Element", the EBML specification does not require any particular storage order for "Elements". The Matroska specification however defines mandates and recommendations for ordering certain "Elements" in order to facilitate better playback, seeking, and editing efficiency. This section describes and offers rationale for ordering requirements and recommendations for Matroska.

11.1. Top-Level Elements

The "Info Element" is the only REQUIRED "Top-Level Element" in a Matroska file. To be playable, Matroska MUST also contain at least one "Tracks Element" and "Cluster Element". The first "Info Element" and the first "Tracks Element" MUST either be stored before the first "Cluster Element" or both SHALL be referenced by a "SeekHead Element" occurring before the first "Cluster Element".

It is possible to edit a Matroska file after it has been created. For example, chapters, tags, or attachments can be added. When new "Top-Level Elements" are added to a Matroska file, the "SeekHead" Element(s) MUST be updated so that the "SeekHead" Element(s) itemize the identity and position of all "Top-Level Elements". Editing, removing, or adding "Elements" to a Matroska file often requires that some existing "Elements" be voided or extended; therefore, it is RECOMMENDED to use "Void Elements" as padding in between "Top-Level Elements".

11.2. CRC-32

As noted by the EBML specification, if a "CRC-32 Element" is used, then the "CRC-32 Element" MUST be the first ordered "Element" within its "Parent Element". The Matroska specification recommends that "CRC-32 Elements" SHOULD NOT be used as an immediate "Child Element" of the "Segment Element"; however all "Top-Level Elements" of an "EBML Document" SHOULD include a "CRC-32 Element" as a "Child Element".

11.3. SeekHead

If used, the first "SeekHead Element" SHOULD be the first non-"CRC-32 Child Element" of the "Segment Element". If a second "SeekHead Element" is used, then the first "SeekHead Element" MUST reference the identity and position of the second "SeekHead". Additionally, the second "SeekHead Element" MUST only reference "Cluster" Elements and not any other "Top-Level Element" already contained within the first "SeekHead Element". The second "SeekHead Element" MAY be

stored in any order relative to the other "Top-Level Elements". Whether one or two "SeekHead Element(s)" are used, the "SeekHead Element(s)" MUST collectively reference the identity and position of all "Top-Level Elements" except for the first "SeekHead Element".

It is RECOMMENDED that the first "SeekHead Element" be followed by a "Void Element" to allow for the "SeekHead Element" to be expanded to cover new "Top-Level Elements" that could be added to the Matroska file, such as "Tags", "Chapters", and "Attachments Elements".

11.4. Cues (index)

The "Cues Element" is RECOMMENDED to optimize seeking access in Matroska. It is programmatically simpler to add the "Cues Element" after all "Cluster Elements" have been written because this does not require a prediction of how much space to reserve before writing the "Cluster Elements". However, storing the "Cues Element" before the "Cluster Elements" can provide some seeking advantages. If the "Cues Element" is present, then it SHOULD either be stored before the first "Cluster Element" or be referenced by a "SeekHead Element".

11.5. Info

The first "Info Element" SHOULD occur before the first "Tracks Element" and first "Cluster Element" except when referenced by a "SeekHead Element".

11.6. Chapters Element

The "Chapters Element" SHOULD be placed before the "Cluster Element(s)". The "Chapters Element" can be used during playback even if the user does not need to seek. It immediately gives the user information about what section is being read and what other sections are available. In the case of Ordered Chapters it is RECOMMENDED to evaluate the logical linking even before playing. The "Chapters Element" SHOULD be placed before the first "Tracks Element" and after the first "Info Element".

11.7. Attachments

The "Attachments Element" is not intended to be used by default when playing the file, but could contain information relevant to the content, such as cover art or fonts. Cover art is useful even before the file is played and fonts could be needed before playback starts for initialization of subtitles. The "Attachments Element" MAY be placed before the first "Cluster Element"; however if the "Attachments Element" is likely to be edited, then it SHOULD be placed after the last "Cluster Element".

11.8. Tags

The "Tags Element" is most subject to changes after the file was originally created. For easier editing, the "Tags Element" SHOULD be placed at the end of the "Segment Element", even after the "Attachments Element". On the other hand, it is inconvenient to have to seek in the "Segment" for tags, especially for network streams. So it's better if the "Tags Element" is found early in the stream. When editing the "Tags Element", the original "Tags Element" at the beginning can be overwritten with a "Void Element" and a new "Tags Element" written at the end of the "Segment Element". The file size will only marginally change.

11.9. Optimum layout from a muxer

* SeekHead
* Info
* Tracks
* Chapters
* Attachments
* Tags
* Clusters
* Cues

11.10. Optimum layout after editing tags

* SeekHead
* Info
* Tracks
* Chapters
* Attachments
* Void
* Clusters
* Cues

* Tags

11.11. Optimum layout with Cues at the front

* SeekHead
* Info
* Tracks
* Chapters
* Attachments
* Tags
* Cues
* Clusters

11.12. Cluster Timestamp

The "Timestamp Element" MUST occur as in storage order before any "SimpleBlock", "BlockGroup", or "EncryptedBlock", within the "Cluster Element".

12. Chapters

12.1. Edition and Chapter Flags

12.1.1. Chapter Flags

Two "Chapter Flags" are defined to describe the behavior of the "ChapterAtom Element": "ChapterFlagHidden" and "ChapterFlagEnabled".

If a "ChapterAtom Element" is the "Child Element" of another "ChapterAtom Element" with a "Chapter Flag" set to "true", then the "Child ChapterAtom Element" MUST be interpreted as having its same "Chapter Flag" set to "true". If a "ChapterAtom Element" is the "Child Element" of another "ChapterAtom Element" with a "Chapter Flag" set to "false", or if the "ChapterAtom Element" does not have a "ChapterAtom Element" as its "Parent Element", then it MUST be interpreted according to its own "Chapter Flag".

As an example, consider a "Parent ChapterAtom Element" that has its "ChapterFlagHidden" set to "true" and also contains two child "ChapterAtoms", the first with "ChapterFlagHidden" set to "true" and the second with "ChapterFlagHidden" either set to "false" or not

present at all (in which case the default value of the Element applies, which is "false"). Since the parent "ChapterAtom" has its "ChapterFlagHidden" set to "true", all of its children "ChapterAtoms" MUST also be interpreted as if their "ChapterFlagHidden" is also set to "true". However, if a "Control Track" toggles the parent's "ChapterFlagHidden" flag to "false", then only the parent "ChapterAtom" and its second child "ChapterAtom" MUST be interpreted as if "ChapterFlagHidden" is set to "false". The first child "ChapterAtom", which has the "ChapterFlagHidden" flag set to "true", retains its value until its value is toggled to "false" by a "Control Track".

12.1.2. Edition Flags

Three "Edition Flags" are defined to describe the behavior of the "EditionEntry Element": "EditionFlagHidden", "EditionFlagDefault", and "EditionFlagOrdered".

12.1.2.1. EditionFlagHidden

The "EditionFlagHidden Flag" behaves similarly to the "ChapterFlagHidden Flag": if "EditionFlagHidden" is set to "true", its "Child ChapterAtoms Elements" MUST also be interpreted as if their "ChapterFlagHidden" is also set to "true", regardless of their own "ChapterFlagHidden Flags". If "EditionFlagHidden" is toggled by a "Control Track" to "false", the "ChapterFlagHidden Flags" of the "Child ChapterAtoms Elements" SHALL determine whether the "ChapterAtom" is hidden or not.

12.1.2.2. EditionFlagDefault

It is RECOMMENDED that no more than one "Edition" have an "EditionFlagDefault Flag" set to "true". The first "Edition" with both the "EditionFlagDefault Flag" set to "true" and the "EditionFlagHidden Flag" set to "false" is the Default Edition. When all "EditionFlagDefault Flags" are set to "false", then the first "Edition" with the "EditionFlagHidden Flag" set to "false" is the Default Edition. The Default Edition is the edition that should be used for playback by default.

12.1.2.3. EditionFlagOrdered

The "EditionFlagOrdered Flag" is a significant feature as it enables an "Edition" of "Ordered Chapters" which defines and arranges a virtual timeline rather than simply labeling points within the timeline. For example, with "Editions" of "Ordered Chapters" a single "Matroska file" can present multiple edits of a film without duplicating content. Alternatively, if a videotape is digitized in

full, one "Ordered Edition" could present the full content (including colorbars, countdown, slate, a feature presentation, and black frames), while another "Edition" of "Ordered Chapters" can use "Chapters" that only mark the intended presentation with the colorbars and other ancillary visual information excluded. If an "Edition" of "Ordered Chapters" is enabled, then the "Matroska Player" MUST play those Chapters in their stored order from the timestamp marked in the "ChapterTimeStart Element" to the timestamp marked in to "ChapterTimeEnd Element".

If the "EditionFlagOrdered Flag" is set to "false", "Simple Chapters" are used and only the "ChapterTimeStart" of a "Chapter" is used as chapter mark to jump to the predefined point in the timeline. With "Simple Chapters", a "Matroska Player" MUST ignore certain "Chapter Elements". All these elements are now informational only.

The following list shows the different usage of "Chapter Elements" between an ordered and non-ordered "Edition".

Chapter elements / ordered Edition	False	True
ChapterUID	X	X
ChapterStringUID	X	X
ChapterTimeStart	X	X
ChapterTimeEnd	-	X
ChapterFlagHidden	X	X
ChapterFlagEnabled	X	X
ChapterSegmentUID	-	X
ChapterSegmentEditionUID	-	X
ChapterPhysicalEquiv	X	X
ChapterTrack	-	X
ChapterDisplay	X	X
ChapProcess	-	X

Table 44

Furthermore there are other EBML "Elements" which could be used if the "EditionFlagOrdered Flag" is set to "true".

Other elements / ordered Edition	False	True
Info/SegmentFamily	-	X
Info/ChapterTranslate	-	X
Track/TrackTranslate	-	X

Table 45

These other "Elements" belong to the Matroska DVD menu system and are only used when the "ChapProcessCodecID Element" is set to 1.

12.1.2.3.1. Ordered-Edition and Matroska Segment-Linking

* Hard Linking: "Ordered-Chapters" supersedes the "Hard Linking".

* Soft Linking: In this complex system "Ordered Chapters" are REQUIRED and a "Chapter CODEC" MUST interpret the "ChapProcess" of all chapters.

* Medium Linking: "Ordered Chapters" are used in a normal way and can be combined with the "ChapterSegmentUID" element which establishes a link to another Segment.

See Section 25 on the Linked Segments for more information about "Hard Linking", "Soft Linking", and "Medium Linking".

12.1.3. ChapterSegmentUID

The "ChapterSegmentUID" is a binary value and the base element to set up a "Linked Chapter" in 2 variations: the Linked-Duration linking and the Linked-Edition linking. For both variations, the following 3 conditions MUST be met:

1. The "EditionFlagOrdered Flag" MUST be true.

2. The "ChapterSegmentUID" MUST NOT be the "SegmentUID" of its own "Segment".

3. The linked Segments MUST BE in the same folder.

12.1.3.1. Variation 1: Linked-Duration

Two more conditions MUST be met:

1. "ChapterTimeStart" and "ChapterTimeEnd" timestamps MUST be in the range of the linked Segment duration.

2. "ChapterSegmentEditionUID" MUST be not set.

A "Matroska Player" MUST play the content of the linked Segment from the "ChapterTimeStart" until "ChapterTimeEnd" timestamp.

12.1.3.2. Variation 2: Linked-Edition

When the "ChapterSegmentEditionUID" is set to a valid "EditionUID" from the linked Segment. A "Matroska Player" MUST play these linked "Edition".

12.2. Menu features

The menu features are handled like a _chapter codec_. That means each codec has a type, some private data and some data in the chapters.

The type of the menu system is defined by the "ChapProcessCodecID" parameter. For now, only 2 values are supported : 0 matroska script, 1 menu borrowed from the DVD. he private data depend on the type of menu system (stored in ChapProcessPrivate), idem for the data in the chapters (stored in ChapProcessData).

12.2.1. Matroska Script (0)

This is the case when "ChapProcessCodecID" = 0. This is a script language build for Matroska purposes. The inspiration comes from ActionScript, javascript and other similar scripting languages. The commands are stored as text commands, in UTF-8. The syntax is C like, with commands spanned on many lines, each terminating with a ";". You can also include comments at the end of lines with "//" or comment many lines using "/* */". The scripts are stored in ChapProcessData. For the moment ChapProcessPrivate is not used.

The one and only command existing for the moment is "GotoAndPlay(ChapterUID);". As the same suggests, it means that, when this command is encountered, the "Matroska Player" SHOULD jump to the "Chapter" specified by the UID and play it.

12.2.2. DVD menu (1)

This is the case when "ChapProcessCodecID" = 1. Each level of a chapter corresponds to a logical level in the DVD system that is stored in the first octet of the ChapProcessPrivate. This DVD hierarchy is as follows:

ChapProcessPrivate	DVD Name	Hierarchy	Commands Possible	Comment
0x30	SS	DVD domain	-	First Play, Video Manager, Video Title
0x2A	LU	Language Unit	-	Contains only PGCs
0x28	TT	Title	-	Contains only PGCs
0x20	PGC	Program Group Chain (PGC)	*	
0x18	PG	Program 1 / Program 2 / Program 3	-	
0x10	PTT	Part Of Title 1 / Part Of Title 2	-	Equivalent to the chapters on the sleeve.
0x08	CN	Cell 1 / Cell 2 / Cell 3 / Cell 4 / Cell 5 / Cell 6	-	

Table 46

You can also recover wether a Segment is a Video Manager (VMG), Video Title Set (VTS) or Video Title Set Menu (VTSM) from the ChapterTranslateID element found in the Segment Info. This field uses 2 octets as follows:

1. Domain Type: 0 for VMG, the domain number for VTS and VTSM

2. Domain Value: 0 for VMG and VTSM, 1 for the VTS source.

For instance, the menu part from VTS_01_0.VOB would be coded [1,0] and the content part from VTS_02_3.VOB would be [2,1]. The VMG is always [0,0]

The following octets of ChapProcessPrivate are as follows:

Octet 1	DVD Name	Following Octets
0x30	SS	Domain name code (1: 0x00= First play, 0xC0= VMG, 0x40= VTSM, 0x80= VTS) + VTS(M) number (2)
0x2A	LU	Language code (2) + Language extension (1)
0x28	TT	global Title number (2) + corresponding TTN of the VTS (1)
0x20	PGC	PGC number (2) + Playback Type (1) + Disabled User Operations (4)
0x18	PG	Program number (2)
0x10	PTT	PTT-chapter number (1)
0x08	CN	Cell number [VOB ID(2)][Cell ID(1)][Angle Num(1)]

Table 47

If the level specified in ChapProcessPrivate is a PGC (0x20), there is an octet called the Playback Type, specifying the kind of PGC defined:

* 0x00: entry only/basic PGC

* 0x82: Title+Entry Menu (only found in the Video Manager domain)

* 0x83: Root Menu (only found in the VTSM domain)

* 0x84: Subpicture Menu (only found in the VTSM domain)

* 0x85: Audio Menu (only found in the VTSM domain)

* 0x86: Angle Menu (only found in the VTSM domain)

* 0x87: Chapter Menu (only found in the VTSM domain)

The next 4 following octets correspond to the User Operation flags (http://dvd.sourceforge.net/dvdinfo/uops.html) in the standard PGC. When a bit is set, the command SHOULD be disabled.

ChapProcessData contains the pre/post/cell commands in binary format as there are stored on a DVD. There is just an octet preceding these data to specify the number of commands in the element. As follows: [# of commands(1)][command 1 (8)][command 2 (8)][command 3 (8)].

More information on the DVD commands and format on DVD-replica (http://www.dvd-replica.com/DVD/), where we got most of the info about it. You can also get information on DVD from the DVDinfo project (http://dvd.sourceforge.net/dvdinfo/).

12.3. Example 1 : basic chaptering

In this example a movie is split in different chapters. It could also just be an audio file (album) on which each track corresponds to a chapter.

* 00000ms - 05000ms : Intro

* 05000ms - 25000ms : Before the crime

* 25000ms - 27500ms : The crime

* 27500ms - 38000ms : The killer arrested

* 38000ms - 43000ms : Credits

This would translate in the following matroska form :

```
<Chapters>
  <EditionEntry>
    <EditionUID>16603393396715046047</EditionUID>
    <ChapterAtom>
      <ChapterUID>1193046</ChapterUID>
      <ChapterTimeStart>0</ChapterTimeStart>
```

```
          <ChapterTimeEnd>5000000000</ChapterTimeEnd>
          <ChapterDisplay>
            <ChapString>Intro</ChapString>
            <ChapLanguage>eng</ChapLanguage>
          </ChapterDisplay>
          <ChapterFlagHidden>0</ChapterFlagHidden>
          <ChapterFlagEnabled>1</ChapterFlagEnabled>
        </ChapterAtom>
        <ChapterAtom>
          <ChapterUID>2311527</ChapterUID>
          <ChapterTimeStart>5000000000</ChapterTimeStart>
          <ChapterTimeEnd>25000000000</ChapterTimeEnd>
          <ChapterDisplay>
            <ChapString>Before the crime</ChapString>
            <ChapLanguage>eng</ChapLanguage>
          </ChapterDisplay>
          <ChapterDisplay>
            <ChapString>Avant le crime</ChapString>
            <ChapLanguage>fra</ChapLanguage>
          </ChapterDisplay>
          <ChapterFlagHidden>0</ChapterFlagHidden>
          <ChapterFlagEnabled>1</ChapterFlagEnabled>
        </ChapterAtom>
        <ChapterAtom>
          <ChapterUID>3430008</ChapterUID>
          <ChapterTimeStart>25000000000</ChapterTimeStart>
          <ChapterTimeEnd>27500000000</ChapterTimeEnd>
          <ChapterDisplay>
            <ChapString>The crime</ChapString>
            <ChapLanguage>eng</ChapLanguage>
          </ChapterDisplay>
          <ChapterDisplay>
            <ChapString>Le crime</ChapString>
            <ChapLanguage>fra</ChapLanguage>
          </ChapterDisplay>
          <ChapterFlagHidden>0</ChapterFlagHidden>
          <ChapterFlagEnabled>1</ChapterFlagEnabled>
        </ChapterAtom>
        <ChapterAtom>
          <ChapterUID>4548489</ChapterUID>
          <ChapterTimeStart>27500000000</ChapterTimeStart>
          <ChapterTimeEnd>38000000000</ChapterTimeEnd>
          <ChapterDisplay>
            <ChapString>After the crime</ChapString>
            <ChapLanguage>eng</ChapLanguage>
          </ChapterDisplay>
          <ChapterDisplay>
            <ChapString>Après le crime</ChapString>
```

```
          <ChapLanguage>fra</ChapLanguage>
        </ChapterDisplay>
        <ChapterFlagHidden>0</ChapterFlagHidden>
        <ChapterFlagEnabled>1</ChapterFlagEnabled>
      </ChapterAtom>
      <ChapterAtom>
        <ChapterUID>5666960</ChapterUID>
        <ChapterTimeStart>38000000000</ChapterTimeStart>
        <ChapterTimeEnd>43000000000</ChapterTimeEnd>
        <ChapterDisplay>
          <ChapString>Credits</ChapString>
          <ChapLanguage>eng</ChapLanguage>
        </ChapterDisplay>
        <ChapterDisplay>
          <ChapString>Générique</ChapString>
          <ChapLanguage>fra</ChapLanguage>
        </ChapterDisplay>
        <ChapterFlagHidden>0</ChapterFlagHidden>
        <ChapterFlagEnabled>1</ChapterFlagEnabled>
      </ChapterAtom>
      <EditionFlagDefault>0</EditionFlagDefault>
      <EditionFlagHidden>0</EditionFlagHidden>
    </EditionEntry>
  </Chapters>
```

12.4. Example 2 : nested chapters

In this example an (existing) album is split into different chapters, and one of them contain another splitting.

12.4.1. The Micronauts "Bleep To Bleep"

* 00:00 - 12:28 : Baby Wants To Bleep/Rock
  - 00:00 - 04:38 : Baby wants to bleep (pt.1)
  - 04:38 - 07:12 : Baby wants to rock
  - 07:12 - 10:33 : Baby wants to bleep (pt.2)
  - 10:33 - 12:28 : Baby wants to bleep (pt.3)
* 12:30 - 19:38 : Bleeper_O+2
* 19:40 - 22:20 : Baby wants to bleep (pt.4)
* 22:22 - 25:18 : Bleep to bleep

* 25:20 - 33:35 : Baby wants to bleep (k)

* 33:37 - 44:28 : Bleeper

```
<Chapters>
  <EditionEntry>
    <EditionUID>12816908580034014l4</EditionUID>
    <ChapterAtom>
      <ChapterUID>1</ChapterUID>
      <ChapterTimeStart>0</ChapterTimeStart>
      <ChapterTimeEnd>748000000</ChapterTimeEnd>
      <ChapterDisplay>
        <ChapString>Baby wants to Bleep/Rock</ChapString>
        <ChapLanguage>eng</ChapLanguage>
      </ChapterDisplay>
      <ChapterAtom>
        <ChapterUID>2</ChapterUID>
        <ChapterTimeStart>0</ChapterTimeStart>
        <ChapterTimeEnd>278000000</ChapterTimeEnd>
        <ChapterDisplay>
          <ChapString>Baby wants to bleep (pt.1)</ChapString>
          <ChapLanguage>eng</ChapLanguage>
        </ChapterDisplay>
        <ChapterFlagHidden>0</ChapterFlagHidden>
        <ChapterFlagEnabled>1</ChapterFlagEnabled>
      </ChapterAtom>
      <ChapterAtom>
        <ChapterUID>3</ChapterUID>
        <ChapterTimeStart>278000000</ChapterTimeStart>
        <ChapterTimeEnd>432000000</ChapterTimeEnd>
        <ChapterDisplay>
          <ChapString>Baby wants to rock</ChapString>
          <ChapLanguage>eng</ChapLanguage>
        </ChapterDisplay>
        <ChapterFlagHidden>0</ChapterFlagHidden>
        <ChapterFlagEnabled>1</ChapterFlagEnabled>
      </ChapterAtom>
      <ChapterAtom>
        <ChapterUID>4</ChapterUID>
        <ChapterTimeStart>432000000</ChapterTimeStart>
        <ChapterTimeEnd>633000000</ChapterTimeEnd>
        <ChapterDisplay>
          <ChapString>Baby wants to bleep (pt.2)</ChapString>
          <ChapLanguage>eng</ChapLanguage>
        </ChapterDisplay>
        <ChapterFlagHidden>0</ChapterFlagHidden>
        <ChapterFlagEnabled>1</ChapterFlagEnabled>
      </ChapterAtom>
```

```
        <ChapterAtom>
          <ChapterUID>5</ChapterUID>
          <ChapterTimeStart>633000000</ChapterTimeStart>
          <ChapterTimeEnd>748000000</ChapterTimeEnd>
          <ChapterDisplay>
            <ChapString>Baby wants to bleep (pt.3)</ChapString>
            <ChapLanguage>eng</ChapLanguage>
          </ChapterDisplay>
          <ChapterFlagHidden>0</ChapterFlagHidden>
          <ChapterFlagEnabled>1</ChapterFlagEnabled>
        </ChapterAtom>
        <ChapterFlagHidden>0</ChapterFlagHidden>
        <ChapterFlagEnabled>1</ChapterFlagEnabled>
      </ChapterAtom>
      <ChapterAtom>
        <ChapterUID>6</ChapterUID>
        <ChapterTimeStart>750000000</ChapterTimeStart>
        <ChapterTimeEnd>1178500000</ChapterTimeEnd>
        <ChapterDisplay>
          <ChapString>Bleeper_O+2</ChapString>
          <ChapLanguage>eng</ChapLanguage>
        </ChapterDisplay>
        <ChapterFlagHidden>0</ChapterFlagHidden>
        <ChapterFlagEnabled>1</ChapterFlagEnabled>
      </ChapterAtom>
      <ChapterAtom>
        <ChapterUID>7</ChapterUID>
        <ChapterTimeStart>1180500000</ChapterTimeStart>
        <ChapterTimeEnd>1340000000</ChapterTimeEnd>
        <ChapterDisplay>
          <ChapString>Baby wants to bleep (pt.4)</ChapString>
          <ChapLanguage>eng</ChapLanguage>
        </ChapterDisplay>
        <ChapterFlagHidden>0</ChapterFlagHidden>
        <ChapterFlagEnabled>1</ChapterFlagEnabled>
      </ChapterAtom>
      <ChapterAtom>
        <ChapterUID>8</ChapterUID>
        <ChapterTimeStart>1342000000</ChapterTimeStart>
        <ChapterTimeEnd>1518000000</ChapterTimeEnd>
        <ChapterDisplay>
          <ChapString>Bleep to bleep</ChapString>
          <ChapLanguage>eng</ChapLanguage>
        </ChapterDisplay>
        <ChapterFlagHidden>0</ChapterFlagHidden>
        <ChapterFlagEnabled>1</ChapterFlagEnabled>
      </ChapterAtom>
      <ChapterAtom>
```

```
        <ChapterUID>9</ChapterUID>
        <ChapterTimeStart>1520000000</ChapterTimeStart>
        <ChapterTimeEnd>2015000000</ChapterTimeEnd>
        <ChapterDisplay>
          <ChapString>Baby wants to bleep (k)</ChapString>
          <ChapLanguage>eng</ChapLanguage>
        </ChapterDisplay>
        <ChapterFlagHidden>0</ChapterFlagHidden>
        <ChapterFlagEnabled>1</ChapterFlagEnabled>
      </ChapterAtom>
      <ChapterAtom>
        <ChapterUID>10</ChapterUID>
        <ChapterTimeStart>2017000000</ChapterTimeStart>
        <ChapterTimeEnd>2668000000</ChapterTimeEnd>
        <ChapterDisplay>
          <ChapString>Bleeper</ChapString>
          <ChapLanguage>eng</ChapLanguage>
        </ChapterDisplay>
        <ChapterFlagHidden>0</ChapterFlagHidden>
        <ChapterFlagEnabled>1</ChapterFlagEnabled>
      </ChapterAtom>
      <EditionFlagDefault>0</EditionFlagDefault>
      <EditionFlagHidden>0</EditionFlagHidden>
    </EditionEntry>
  </Chapters>
```

13. Attachments

Matroska supports storage of related files and data in the "Attachments Element" (a "Top-Level Element"). "Attachment Elements" can be used to store related cover art, font files, transcripts, reports, error recovery files, picture, or text-based annotations, copies of specifications, or other ancillary files related to the "Segment".

"Matroska Readers" MUST NOT execute files stored as "Attachment Elements".

13.1. Cover Art

This section defines a set of guidelines for the storage of cover art in Matroska files. A "Matroska Reader" MAY use embedded cover art to display a representational still-image depiction of the multimedia contents of the Matroska file.

Only JPEG and PNG image formats SHOULD be used for cover art pictures.

There can be two different covers for a movie/album: a portrait style (e.g., a DVD case) and a landscape style (e.g., a wide banner ad).

There can be two versions of the same cover, the "normal cover" and the "small cover". The dimension of the "normal cover" SHOULD be 600 pixels on the smallest side -- for example, 960x600 for landscape, 600x800 for portrait, or 600x600 for square. The dimension of the "small cover" SHOULD be 120 pixels on the smallest side -- for example, 192x120 or 120x160.

Versions of cover art can be differentiated by the filename, which is stored in the "FileName Element". The default filename of the "normal cover" in square or portrait mode is "cover.(jpg|png)". When stored, the "normal cover" SHOULD be the first Attachment in storage order. The "small cover" SHOULD be prefixed with "small_", such as "small_cover.(jpg|png)". The landscape variant SHOULD be suffixed with "_land", such as "cover_land.(jpg|png)". The filenames are case sensitive.

The following table provides examples of file names for cover art in Attachments.

FileName	Image Orientation	Pixel Length of Smallest Side
cover.jpg	Portrait or square	600
small_cover.png	Portrait or square	120
cover_land.png	Landscape	600
small_cover_land.jpg	Landscape	120

Table 48

14. Cues

The "Cues Element" provides an index of certain "Cluster Elements" to allow for optimized seeking to absolute timestamps within the "Segment". The "Cues Element" contains one or many "CuePoint Elements" which each MUST reference an absolute timestamp (via the "CueTime Element"), a "Track" (via the "CueTrack Element"), and a "Segment Position" (via the "CueClusterPosition Element"). Additional non-mandated Elements are part of the "CuePoint Element"

such as "CueDuration", "CueRelativePosition", "CueCodecState" and others which provide any "Matroska Reader" with additional information to use in the optimization of seeking performance.

14.1. Recommendations

The following recommendations are provided to optimize Matroska performance.

* Unless Matroska is used as a live stream, it SHOULD contain a "Cues Element".

* For each video track, each keyframe SHOULD be referenced by a "CuePoint Element".

* It is RECOMMENDED to not reference non-keyframes of video tracks in "Cues" unless it references a "Cluster Element" which contains a "CodecState Element" but no keyframes.

* For each subtitle track present, each subtitle frame SHOULD be referenced by a "CuePoint Element" with a "CueDuration Element".

* References to audio tracks MAY be skipped in "CuePoint Elements" if a video track is present. When included the "CuePoint Elements" SHOULD reference audio keyframes at most once every 500 milliseconds.

* If the referenced frame is not stored within the first "SimpleBlock", or first "BlockGroup" within its "Cluster Element", then the "CueRelativePosition Element" SHOULD be written to reference where in the "Cluster" the reference frame is stored.

* If a "CuePoint Element" references "Cluster Element" that includes a "CodecState Element", then that "CuePoint Element" MUST use a "CueCodecState Element".

* "CuePoint Elements" SHOULD be numerically sorted in storage order by the value of the "CueTime Element".

15. Matroska Streaming

In Matroska, there are two kinds of streaming: file access and livestreaming.

15.1. File Access

File access can simply be reading a file located on your computer, but also includes accessing a file from an HTTP (web) server or CIFS (Windows share) server. These protocols are usually safe from reading errors and seeking in the stream is possible. However, when a file is stored far away or on a slow server, seeking can be an expensive operation and SHOULD be avoided. The following guidelines, when followed, help reduce the number of seeking operations for regular playback and also have the playback start quickly without a lot of data needed to read first (like a "Cues Element", "Attachment Element" or "SeekHead Element").

Matroska, having a small overhead, is well suited for storing music/videos on file servers without a big impact on the bandwidth used. Matroska does not require the index to be loaded before playing, which allows playback to start very quickly. The index can be loaded only when seeking is requested the first time.

15.2. Livestreaming

Livestreaming is the equivalent of television broadcasting on the internet. There are 2 families of servers for livestreaming: RTP/RTSP and HTTP. Matroska is not meant to be used over RTP. RTP already has timing and channel mechanisms that would be wasted if doubled in Matroska. Additionally, having the same information at the RTP and Matroska level would be a source of confusion if they do not match. Livestreaming of Matroska over HTTP (or any other plain protocol based on TCP) is possible.

A live Matroska stream is different from a file because it usually has no known end (only ending when the client disconnects). For this, all bits of the "size" portion of the "Segment Element" MUST be set to 1. Another option is to concatenate "Segment Elements" with known sizes, one after the other. This solution allows a change of codec/resolution between each segment. For example, this allows for a switch between 4:3 and 16:9 in a television program.

When "Segment Elements" are continuous, certain "Elements", like "MetaSeek", "Cues", "Chapters", and "Attachments", MUST NOT be used.

It is possible for a "Matroska Player" to detect that a stream is not seekable. If the stream has neither a "MetaSeek" list or a "Cues" list at the beginning of the stream, it SHOULD be considered non-seekable. Even though it is possible to seek blindly forward in the stream, it is NOT RECOMMENDED.

In the context of live radio or web TV, it is possible to "tag" the content while it is playing. The "Tags Element" can be placed between "Clusters" each time it is necessary. In that case, the new "Tags Element" MUST reset the previously encountered "Tags Elements" and use the new values instead.

16. Menu Specifications

This document is a _draft of the Menu system_ that will be the default one in "Matroska". As it will just be composed of a Control Track, it will be seen as a "codec" and could be replaced later by something else if needed.

A menu is like what you see on DVDs, when you have some screens to select the audio format, subtitles or scene selection.

16.1. Requirements

What we'll try to have is a system that can do almost everything done on a DVD, or more, or better, or drop the unused features if necessary.

As the name suggests, a Control Track is a track that can control the playback of the file and/or all the playback features. To make it as simple as possible for "Matroska Players", the Control Track will just give orders to the "Matroska Player" and get the actions associated with the highlights/hotspots.

16.1.1. Highlights/Hotspots

A highlight is basically a rectangle/key associated with an action UID. When that rectangle/key is activated, the "Matroska Player" send the UID of the action to the Control Track handler (codec). The fact that it can also be a key means that even for audio only files, a keyboard shortcut or button panel could be used for menus. But in that case, the hotspot will have to be associated with a name to display.

This highlight is sent from the Control Track to the "Matroska Player". Then the "Matroska Player" has to handle that highlight until it's deactivated; see Section 16.1.2.

The highlight contains a UID of the action, a displayable name (UTF-8), an associated key (list of keys to be defined, probably up/down/left/right/select), a screen position/range and an image to display. The image will be displayed either when the user place the mouse over the rectangle (or any other shape), or when an option of the screen is selected (not activated). There could be a second

image used when the option is activated. And there could be a third image that can serve as background. This way you could have a still image (like in some DVDs) for the menu and behind that image blank video (small bitrate).

When a highlight is activated by the user, the "Matroska Player" has to send the UID of the action to the Control Track. Then the Control Track codec will handle the action and possibly give new orders to the "Matroska Player".

The format used for storing images SHOULD be extensible. For the moment we'll use PNG and BMP, both with alpha channel.

16.1.2. Playback features

All the following features will be sent from the Control Track to the "Matroska Player" :

* Jump to chapter (UID, prev, next, number)
* Disable all tracks of a kind (audio, video, subtitle)
* Enable track UID (the kind doesn't matter)
* Define/Disable a highlight
* Enable/Disable jumping
* Enable/Disable track selection of a kind
* Select Edition ID (see chapters)
* Pause playback
* Stop playback
* Enable/Disable a Chapter UID
* Hide/Unhide a Chapter UID

All the actions will be written in a normal Matroska track, with a timestamp. A "Menu Frame" SHOULD be able to contain more that one action/highlight for a given timestamp. (to be determined, EBML format structure)

16.1.3. Player requirements

Some "Matroska Players" might not support the control track. That mean they will play the active/looped parts as part of the data. So I suggest putting the active/looped parts of a movie at the end of a movie. When a Menu-aware "Matroska Player" encounter the default Control Track of a "Matroska" file, the first order SHOULD be to jump at the start of the active/looped part of the movie.

16.2. Working Graph

```
Matroska Source file -> Control Track <-> Player.
                     -> other tracks   -> rendered
```

17. Unknown elements

Matroska is based upon the principle that a reading application does not have to support 100% of the specifications in order to be able to play the file. A Matroska file therefore contains version indicators that tell a reading application what to expect.

It is possible and valid to have the version fields indicate that the file contains Matroska "Elements" from a higher specification version number while signaling that a reading application MUST only support a lower version number properly in order to play it back (possibly with a reduced feature set). For example, a reading application supporting at least Matroska version "V" reading a file whose "DocTypeReadVersion" field is equal to or lower than "V" MUST skip Matroska/EBML "Elements" it encounters but does not know about if that unknown element fits into the size constraints set by the current "Parent Element".

18. Default Values

The default value of an "Element" is assumed when not present in the data stream. It is assumed only in the scope of its "Parent Element". For example, the "Language Element" is in the scope of the "Track Element". If the "Parent Element" is not present or assumed, then the "Child Element" cannot be assumed.

19. DefaultDecodedFieldDuration

The "DefaultDecodedFieldDuration Element" can signal to the displaying application how often fields of a video sequence will be available for displaying. It can be used for both interlaced and progressive content. If the video sequence is signaled as interlaced, then the period between two successive fields at the output of the decoding process equals "DefaultDecodedFieldDuration".

For video sequences signaled as progressive, it is twice the value of "DefaultDecodedFieldDuration".

These values are valid at the end of the decoding process before post-processing (such as deinterlacing or inverse telecine) is applied.

Examples:

* Blu-ray movie: 1000000000ns/(48/1.001) = 20854167ns
* PAL broadcast/DVD: 1000000000ns/(50/1.000) = 20000000ns
* N/ATSC broadcast: 1000000000ns/(60/1.001) = 16683333ns
* hard-telecined DVD: 1000000000ns/(60/1.001) = 16683333ns (60 encoded interlaced fields per second)
* soft-telecined DVD: 1000000000ns/(60/1.001) = 16683333ns (48 encoded interlaced fields per second, with "repeat_first_field = 1")

20. Encryption

Encryption in Matroska is designed in a very generic style to allow people to implement whatever form of encryption is best for them. It is possible to use the encryption framework in Matroska as a type of DRM (Digital Rights Management).

Because encryption occurs within the "Block Element", it is possible to manipulate encrypted streams without decrypting them. The streams could potentially be copied, deleted, cut, appended, or any number of other possible editing techniques without decryption. The data can be used without having to expose it or go through the decrypting process.

Encryption can also be layered within Matroska. This means that two completely different types of encryption can be used, requiring two separate keys to be able to decrypt a stream.

Encryption information is stored in the "ContentEncodings Element" under the "ContentEncryption Element".

21. Image Presentation

21.1. Cropping

The "PixelCrop Elements" ("PixelCropTop", "PixelCropBottom", "PixelCropRight", and "PixelCropLeft") indicate when, and by how much, encoded videos frames SHOULD be cropped for display. These Elements allow edges of the frame that are not intended for display, such as the sprockets of a full-frame film scan or the VANC area of a digitized analog videotape, to be stored but hidden. "PixelCropTop" and "PixelCropBottom" store an integer of how many rows of pixels SHOULD be cropped from the top and bottom of the image (respectively). "PixelCropLeft" and "PixelCropRight" store an integer of how many columns of pixels SHOULD be cropped from the left and right of the image (respectively). For example, a pillar-boxed video that stores a 1440x1080 visual image within the center of a padded 1920x1080 encoded image MAY set both "PixelCropLeft" and "PixelCropRight" to "240", so that a "Matroska Player" SHOULD crop off 240 columns of pixels from the left and right of the encoded image to present the image with the pillar-boxes hidden.

21.2. Rotation

The ProjectionPoseRoll Element (see Section 10.4.1.31.20.5) can be used to indicate that the image from the associated video track SHOULD be rotated for presentation. For instance, the following representation of the Projection Element Section 10.4.1.31.20) and the ProjectionPoseRoll Element represents a video track where the image SHOULD be presentation with a 90 degree counter-clockwise rotation.

```
<Projection>
  <ProjectionPoseRoll>90</ProjectionPoseRoll>
</Projection>
```

22. Matroska versioning

The "EBML Header" of each Matroska document informs the reading application on what version of Matroska to expect. The "Elements" within "EBML Header" with jurisdiction over this information are "DocTypeVersion" and "DocTypeReadVersion".

"DocTypeVersion" MUST be equal to or greater than the highest Matroska version number of any "Element" present in the Matroska file. For example, a file using the "SimpleBlock Element" MUST have a "DocTypeVersion" equal to or greater than 2. A file containing "CueRelativePosition" Elements MUST have a "DocTypeVersion" equal to or greater than 4.

The "DocTypeReadVersion" MUST contain the minimum version number that a reading application can minimally support in order to play the file back -- optionally with a reduced feature set. For example, if a file contains only "Elements" of version 2 or lower except for "CueRelativePosition" (which is a version 4 Matroska "Element"), then "DocTypeReadVersion" SHOULD still be set to 2 and not 4 because evaluating "CueRelativePosition" is not necessary for standard playback -- it makes seeking more precise if used.

"DocTypeVersion" MUST always be equal to or greater than "DocTypeReadVersion".

A reading application supporting Matroska version "V" MUST NOT refuse to read an application with "DocReadTypeVersion" equal to or lower than "V" even if "DocTypeVersion" is greater than "V". See also the note about Unknown Elements in Section 17.

23. MIME Types

There is no IETF endorsed MIME type for Matroska files. These definitions can be used:

* .mka : Matroska audio "audio/x-matroska"

* .mkv : Matroska video "video/x-matroska"

* .mk3d : Matroska 3D video "video/x-matroska-3d"

24. Segment Position

The "Segment Position" of an "Element" refers to the position of the first octet of the "Element ID" of that "Element", measured in octets, from the beginning of the "Element Data" section of the containing "Segment Element". In other words, the "Segment Position" of an "Element" is the distance in octets from the beginning of its containing "Segment Element" minus the size of the "Element ID" and "Element Data Size" of that "Segment Element". The "Segment Position" of the first "Child Element" of the "Segment Element" is 0. An "Element" which is not stored within a "Segment Element", such as the "Elements" of the "EBML Header", do not have a "Segment Position".

24.1. Segment Position Exception

"Elements" that are defined to store a "Segment Position" MAY define reserved values to indicate a special meaning.

24.2. Example of Segment Position

This table presents an example of "Segment Position" by showing a hexadecimal representation of a very small Matroska file with labels to show the offsets in octets. The file contains a "Segment Element" with an "Element ID" of "0x18538067" and a "MuxingApp Element" with an "Element ID" of "0x4D80".

```
     0                               1                               2
     0  1  2  3  4  5  6  7  8  9  0  1  2  3  4  5  6  7  8  9  0  1
     +--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+--+
   0 |1A|45|DF|A3|8B|42|82|88|6D|61|74|72|6F|73|6B|61|18|53|80|67|
  20 |93|15|49|A9|66|8E|4D|80|84|69|65|74|66|57|41|84|69|65|74|66|
```

In the above example, the "Element ID" of the "Segment Element" is stored at offset 16, the "Element Data Size" of the "Segment Element" is stored at offset 20, and the "Element Data" of the "Segment Element" is stored at offset 21.

The "MuxingApp Element" is stored at offset 26. Since the "Segment Position" of an "Element" is calculated by subtracting the position of the "Element Data" of the containing "Segment Element" from the position of that "Element", the "Segment Position" of "MuxingApp Element" in the above example is ’26 - 21’ or ’5’.

25. Linked Segments

Matroska provides several methods to link two or many "Segment Elements" together to create a "Linked Segment". A "Linked Segment" is a set of multiple "Segments" related together into a single presentation by using Hard Linking, Medium Linking, or Soft Linking. All "Segments" within a "Linked Segment" MUST utilize the same track numbers and timescale. All "Segments" within a "Linked Segment" MUST be stored within the same directory. All "Segments" within a "Linked Segment" MUST store a "SegmentUID".

25.1. Hard Linking

Hard Linking (also called splitting) is the process of creating a "Linked Segment" by relating multiple "Segment Elements" using the "NextUID" and "PrevUID" Elements. Within a "Linked Segment", the timestamps of each "Segment" MUST follow consecutively in linking order. With Hard Linking, the chapters of any "Segment" within the "Linked Segment" MUST only reference the current "Segment". With Hard Linking, the "NextUID" and "PrevUID" MUST reference the respective "SegmentUID" values of the next and previous "Segments". The first "Segment" of a "Linked Segment" SHOULD have a "NextUID Element" and MUST NOT have a "PrevUID Element". The last "Segment"

of a "Linked Segment" SHOULD have a "PrevUID Element" and MUST NOT have a "NextUID Element". The middle "Segments" of a "Linked Segment" SHOULD have both a "NextUID Element" and a "PrevUID Element".

In a chain of "Linked Segments" the "NextUID" always takes precedence over the "PrevUID". So if SegmentA has a NextUID to SegmentB and SegmentB has a PrevUID to SegmentC, the link to use is SegmentA to SegmentB. If SegmentB has a PrevUID to SegmentA but SegmentA has no NextUID, then the Matroska Player MAY consider these two Segments linked as SegmentA followed by SegmentB.

As an example, three "Segments" can be Hard Linked as a "Linked Segment" through cross-referencing each other with "SegmentUID", "PrevUID", and "NextUID", as in this table.

file name	SegmentUID	PrevUID	NextUID
start.mkv	71000c23cd310998 53fbc94dd984a5dd	n/a	a77b3598941cb803 eac0fcdafe44fac9
middle.mkv	a77b3598941cb803 eac0fcdafe44fac9	71000c23cd310998 53fbc94dd984a5dd	6c92285fa6d3e827 b198d120ea3ac674
end.mkv	6c92285fa6d3e827 b198d120ea3ac674	a77b3598941cb803 eac0fcdafe44fac9	n/a

Table 49

An other example where only the "NextUID" Element is used.

file name	SegmentUID	PrevUID	NextUID
start.mkv	71000c23cd310998 53fbc94dd984a5dd	n/a	a77b3598941cb803 eac0fcdafe44fac9
middle.mkv	a77b3598941cb803 eac0fcdafe44fac9	n/a	6c92285fa6d3e827 b198d120ea3ac674
end.mkv	6c92285fa6d3e827 b198d120ea3ac674	n/a	n/a

Table 50

A next example where only the "PrevUID" Element is used.

file name	SegmentUID	PrevUID	NextUID
start.mkv	71000c23cd310998 53fbc94dd984a5dd	n/a	n/a
middle.mkv	a77b3598941cb803 eac0fcdafe44fac9	71000c23cd310998 53fbc94dd984a5dd	n/a
end.mkv	6c92285fa6d3e827 b198d120ea3ac674	a77b3598941cb803 eac0fcdafe44fac9	n/a

Table 51

In this example only the "middle.mkv" is using the "PrevUID" and "NextUID" Elements.

file name	SegmentUID	PrevUID	NextUID
start.mkv	71000c23cd310998 53fbc94dd984a5dd	n/a	n/a
middle.mkv	a77b3598941cb803 eac0fcdafe44fac9	71000c23cd310998 53fbc94dd984a5dd	6c92285fa6d3e827 b198d120ea3ac674
end.mkv	6c92285fa6d3e827 b198d120ea3ac674	n/a	n/a

Table 52

25.2. Medium Linking

Medium Linking creates relationships between "Segments" using Ordered Chapters and the "ChapterSegmentUID Element". A "Segment Edition" with Ordered Chapters MAY contain "Chapter Elements" that reference timestamp ranges from other "Segments". The "Segment" referenced by the Ordered Chapter via the "ChapterSegmentUID Element" SHOULD be played as part of a Linked Segment. The timestamps of Segment content referenced by Ordered Chapters MUST be adjusted according to the cumulative duration of the the previous Ordered Chapters.

As an example a file named "intro.mkv" could have a "SegmentUID" of "0xb16a58609fc7e60653a60c984fc11ead". Another file called "program.mkv" could use a Chapter Edition that contains two Ordered Chapters. The first chapter references the "Segment" of "intro.mkv" with the use of a "ChapterSegmentUID", "ChapterSegmentEditionUID", "ChapterTimeStart", and optionally a "ChapterTimeEnd" element. The second chapter references content within the "Segment" of "program.mkv". A "Matroska Player" SHOULD recognize the "Linked Segment" created by the use of "ChapterSegmentUID" in an enabled "Edition" and present the reference content of the two "Segments" together.

25.3. Soft Linking

Soft Linking is used by codec chapters. They can reference another "Segment" and jump to that "Segment". The way the "Segments" are described are internal to the chapter codec and unknown to the Matroska level. But there are "Elements" within the "Info Element" (such as "ChapterTranslate") that can translate a value representing a "Segment" in the chapter codec and to the current "SegmentUID". All "Segments" that could be used in a "Linked Segment" in this way SHOULD be marked as members of the same family via the "SegmentFamily Element", so that the "Matroska Player" can quickly switch from one to the other.

26. Track Flags

26.1. Default flag

The "default track" flag is a hint for a "Matroska Player" and SHOULD always be changeable by the user. If the user wants to see or hear a track of a certain kind (audio, video, subtitles) and hasn’t chosen a specific track, the "Matroska Player" SHOULD use the first track of that kind whose "default track" flag is set to "1". If no such track is found, then the first track of this kind SHOULD be chosen.

Only one track of a kind MAY have its "default track" flag set in a segment. If a track entry does not contain the "default track" flag element, then its default value "1" is to be used.

26.2. Forced flag

The "forced" flag tells the "Matroska Player" that it MUST display/ play this track or another track of the same kind that also has its "forced" flag set. When there are multiple "forced" tracks, the "Matroska Player" SHOULD determine the track based upon the language of the forced flag or use the default flag if no track matches the use languages. Another track of the same kind without the "forced"

flag may be use simultaneously with the "forced" track, like DVD subtitles.

26.3. Track Operation

"TrackOperation" allows combining multiple tracks to make a virtual one. It uses two separate system to combine tracks. One to create a 3D "composition" (left/right/background planes) and one to simplify join two tracks together to make a single track.

A track created with "TrackOperation" is a proper track with a UID and all its flags. However the codec ID is meaningless because each "sub" track needs to be decoded by its own decoder before the "operation" is applied. The "Cues Elements" corresponding to such a virtual track SHOULD be the sum of the "Cues Elements" for each of the tracks it's composed of (when the "Cues" are defined per track).

In the case of "TrackJoinBlocks", the "Block Elements" (from "BlockGroup" and "SimpleBlock") of all the tracks SHOULD be used as if they were defined for this new virtual "Track". When two "Block Elements" have overlapping start or end timestamps, it's up to the underlying system to either drop some of these frames or render them the way they overlap. This situation SHOULD be avoided when creating such tracks as you can never be sure of the end result on different platforms.

26.4. Overlay Track

Overlay tracks SHOULD be rendered in the same channel as the track its linked to. When content is found in such a track, it SHOULD be played on the rendering channel instead of the original track.

26.5. Multi-planar and 3D videos

There are two different ways to compress 3D videos: have each eye track in a separate track and have one track have both eyes combined inside (which is more efficient, compression-wise). Matroska supports both ways.

For the single track variant, there is the "StereoMode Element", which defines how planes are assembled in the track (mono or left-right combined). Odd values of StereoMode means the left plane comes first for more convenient reading. The pixel count of the track ("PixelWidth"/"PixelHeight") is the raw amount of pixels, for example 3840x1080 for full HD side by side, and the "DisplayWidth"/"DisplayHeight" in pixels is the amount of pixels for one plane (1920x1080 for that full HD stream). Old stereo 3D were displayed using anaglyph (cyan and red colours separated). For compatibility with such movies, there is a value of the StereoMode that corresponds to AnaGlyph.

There is also a "packed" mode (values 13 and 14) which consists of packing two frames together in a "Block" using lacing. The first frame is the left eye and the other frame is the right eye (or vice versa). The frames SHOULD be decoded in that order and are possibly dependent on each other (P and B frames).

For separate tracks, Matroska needs to define exactly which track does what. "TrackOperation" with "TrackCombinePlanes" do that. For more details look at Section 26.3 on how TrackOperation works.

The 3D support is still in infancy and may evolve to support more features.

The StereoMode used to be part of Matroska v2 but it didn’t meet the requirement for multiple tracks. There was also a bug in libmatroska prior to 0.9.0 that would save/read it as 0x53B9 instead of 0x53B8. "Matroska Readers" may support these legacy files by checking Matroska v2 or 0x53B9. The older values were 0: mono, 1: right eye, 2: left eye, 3: both eyes.

27. Timestamps

Historically timestamps in Matroska were mistakenly called timecodes. The "Timestamp Element" was called Timecode, the "TimestampScale Element" was called TimecodeScale, the "TrackTimestampScale Element" was called TrackTimecodeScale and the "ReferenceTimestamp Element" was called ReferenceTimeCode.

27.1. Timestamp Types

* Absolute Timestamp = Block+Cluster

* Relative Timestamp = Block

* Scaled Timestamp = Block+Cluster

* Raw Timestamp = (Block+Cluster)*TimestampScale*TrackTimestampScale

27.2. Block Timestamps

The "Block Element"'s timestamp MUST be a signed integer that represents the "Raw Timestamp" relative to the "Cluster"'s "Timestamp Element", multiplied by the "TimestampScale Element". See Section 27.4 for more information.

The "Block Element"'s timestamp MUST be represented by a 16bit signed integer (sint16). The "Block"'s timestamp has a range of -32768 to +32767 units. When using the default value of the "TimestampScale Element", each integer represents 1ms. The maximum time span of "Block Elements" in a "Cluster" using the default "TimestampScale Element" of 1ms is 65536ms.

If a "Cluster"'s "Timestamp Element" is set to zero, it is possible to have "Block Elements" with a negative "Raw Timestamp". "Block Elements" with a negative "Raw Timestamp" are not valid.

27.3. Raw Timestamp

The exact time of an object SHOULD be represented in nanoseconds. To find out a "Block"'s "Raw Timestamp", you need the "Block"'s "Timestamp Element", the "Cluster"'s "Timestamp Element", and the "TimestampScale Element".

27.4. TimestampScale

The "TimestampScale Element" is used to calculate the "Raw Timestamp" of a "Block". The timestamp is obtained by adding the "Block"'s timestamp to the "Cluster"'s "Timestamp Element", and then multiplying that result by the "TimestampScale". The result will be the "Block"'s "Raw Timestamp" in nanoseconds. The formula for this would look like:

(a + b) * c

a = `Block`'s Timestamp
b = `Cluster`'s Timestamp
c = `TimestampScale`

For example, assume a "Cluster"'s "Timestamp" has a value of 564264, the "Block" has a "Timestamp" of 1233, and the "TimestampScale Element" is the default of 1000000.

(1233 + 564264) * 1000000 = 565497000000

So, the "Block" in this example has a specific time of 565497000000 in nanoseconds. In milliseconds this would be 565497ms.

27.5. TimestampScale Rounding

Because the default value of "TimestampScale" is 1000000, which makes each integer in the "Cluster" and "Block" "Timestamp Elements" equal 1ms, this is the most commonly used. When dealing with audio, this causes inaccuracy when seeking. When the audio is combined with video, this is not an issue. For most cases, the the synch of audio to video does not need to be more than 1ms accurate. This becomes obvious when one considers that sound will take 2-3ms to travel a single meter, so distance from your speakers will have a greater effect on audio/visual synch than this.

However, when dealing with audio-only files, seeking accuracy can become critical. For instance, when storing a whole CD in a single track, a user will want to be able to seek to the exact sample that a song begins at. If seeking a few sample ahead or behind, a crack or pop may result as a few odd samples are rendered. Also, when performing precise editing, it may be very useful to have the audio accuracy down to a single sample.

When storing timestamps for an audio stream, the "TimestampScale Element" SHOULD have an accuracy of at least that of the audio sample rate, otherwise there are rounding errors that prevent users from knowing the precise location of a sample. Here’s how a program has to round each timestamp in order to be able to recreate the sample number accurately.

Let’s assume that the application has an audio track with a sample rate of 44100. As written above the "TimestampScale" MUST have at least the accuracy of the sample rate itself: 1000000000 / 44100 = 22675.7369614512. This value MUST always be truncated. Otherwise the accuracy will not suffice. So in this example the application will use 22675 for the "TimestampScale". The application could even use some lower value like 22674, which would allow it to be a little bit imprecise about the original timestamps. But more about that in a minute.

Next the application wants to write sample number 52340 and calculates the timestamp. This is easy. In order to calculate the "Raw Timestamp" in ns all it has to do is calculate "Raw Timestamp = round(1000000000 * sample_number / sample_rate)". Rounding at this stage is very important! The application might skip it if it choses a slightly smaller value for the "TimestampScale" factor instead of the truncated one like shown above. Otherwise it has to round or the results won't be reversible. For our example we get "Raw Timestamp = round(1000000000 * 52340 / 44100) = round(1186848072.56236) = 1186848073".

The next step is to calculate the "Absolute Timestamp" - that is the timestamp that will be stored in the Matroska file. Here the application has to divide the "Raw Timestamp" from the previous paragraph by the "TimestampScale" factor and round the result: "Absolute Timestamp = round(Raw Timestamp / TimestampScale_factor)", which will result in the following for our example: "Absolute Timestamp = round(1186848073 / 22675) = round(52341.7011245866) = 52342". This number is the one the application has to write to the file.

Now our file is complete, and we want to play it back with another application. Its task is to find out which sample the first application wrote into the file. So it starts reading the Matroska file and finds the "TimestampScale" factor 22675 and the audio sample rate 44100. Later it finds a data block with the "Absolute Timestamp" of 52342. But how does it get the sample number from these numbers?

First it has to calculate the "Raw Timestamp" of the block it has just read. Here's no rounding involved, just an integer multiplication: "Raw Timestamp = Absolute Timestamp * TimestampScale_factor". In our example: "Raw Timestamp = 52342 * 22675 = 1186854850".

The conversion from the "Raw Timestamp" to the sample number again requires rounding: "sample_number = round(Raw Timestamp * sample_rate / 1000000000)". In our example: "sample_number = round(1186854850 * 44100 / 1000000000) = round(52340.298885) = 52340". This is exactly the sample number that the previous program started with.

Some general notes for a program:

1. Always calculate the timestamps / sample numbers with floating point numbers of at least 64bit precision (called 'double' in most modern programming languages). If you're calculating with integers, then make sure they're 64bit long, too.

2. Always round if you divide. Always! If you don’t you’ll end up with situations in which you have a timestamp in the Matroska file that does not correspond to the sample number that it started with. Using a slightly lower timestamp scale factor can help here in that it removes the need for proper rounding in the conversion from sample number to "Raw Timestamp".

27.6. TrackTimestampScale

The "TrackTimestampScale Element" is used align tracks that would otherwise be played at different speeds. An example of this would be if you have a film that was originally recorded at 24fps video. When playing this back through a PAL broadcasting system, it is standard to speed up the film to 25fps to match the 25fps display speed of the PAL broadcasting standard. However, when broadcasting the video through NTSC, it is typical to leave the film at its original speed. If you wanted to make a single file where there was one video stream, and an audio stream used from the PAL broadcast, as well as an audio stream used from the NTSC broadcast, you would have the problem that the PAL audio stream would be 1/24th faster than the NTSC audio stream, quickly leading to problems. It is possible to stretch out the PAL audio track and re-encode it at a slower speed, however when dealing with lossy audio codecs, this often results in a loss of audio quality and/or larger file sizes.

This is the type of problem that "TrackTimestampScale" was designed to fix. Using it, the video can be played back at a speed that will synch with either the NTSC or the PAL audio stream, depending on which is being used for playback. To continue the above example:

Track 1: Video
Track 2: NTSC Audio
Track 3: PAL Audio

Because the NTSC track is at the original speed, it will used as the default value of 1.0 for its "TrackTimestampScale". The video will also be aligned to the NTSC track with the default value of 1.0.

The "TrackTimestampScale" value to use for the PAL track would be calculated by determining how much faster the PAL track is than the NTSC track. In this case, because we know the video for the NTSC audio is being played back at 24fps and the video for the PAL audio is being played back at 25fps, the calculation would be:

25/24 is almost 1.0416666666666666667

When writing a file that uses a non-default "TrackTimestampScale", the values of the "Block"'s timestamp are whatever they would be when normally storing the track with a default value for the "TrackTimestampScale". However, the data is interleaved a little differently. Data SHOULD be interleaved by its Raw Timestamp, see Section 27.3, in the order handed back from the encoder. The "Raw Timestamp" of a "Block" from a track using "TrackTimestampScale" is calculated using:

"(Block's Timestamp + Cluster's Timestamp) * TimestampScale * TrackTimestampScale"

So, a Block from the PAL track above that had a Scaled Timestamp, see Section 27.1, of 100 seconds would have a "Raw Timestamp" of 104.66666667 seconds, and so would be stored in that part of the file.

When playing back a track using the "TrackTimestampScale", if the track is being played by itself, there is no need to scale it. From the above example, when playing the Video with the NTSC Audio, neither are scaled. However, when playing back the Video with the PAL Audio, the timestamps from the PAL Audio track are scaled using the "TrackTimestampScale", resulting in the video playing back in synch with the audio.

It would be possible for a "Matroska Player" to also adjust the audio's samplerate at the same time as adjusting the timestamps if you wanted to play the two audio streams synchronously. It would also be possible to adjust the video to match the audio's speed. However, for playback, the selected track(s) timestamps SHOULD be adjusted if they need to be scaled.

While the above example deals specifically with audio tracks, this element can be used to align video, audio, subtitles, or any other type of track contained in a Matroska file.

28. Normative References

[I-D.ietf-cellar-codec]
Lhomme, S., Bunkus, M., and D. Rice, "Matroska Media Container Codec Specifications", Work in Progress, Internet-Draft, draft-ietf-cellar-codec-04, 17 April 2020, <https://tools.ietf.org/html/draft-ietf-cellar-codec-04>.

[I-D.ietf-cellar-tags]
Lhomme, S., Bunkus, M., and D. Rice, "Matroska Media Container Tag Specifications", Work in Progress, Internet-Draft, draft-ietf-cellar-tags-04, 17 April 2020, <https://tools.ietf.org/html/draft-ietf-cellar-tags-04>.

[RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, DOI 10.17487/RFC2119, March 1997, <https://www.rfc-editor.org/info/rfc2119>.

[RFC8174] Leiba, B., "Ambiguity of Uppercase vs Lowercase in RFC 2119 Key Words", BCP 14, RFC 8174, DOI 10.17487/RFC8174, May 2017, <https://www.rfc-editor.org/info/rfc8174>.

[RFC8794] Lhomme, S., Rice, D., and M. Bunkus, "Extensible Binary Meta Language", RFC 8794, DOI 10.17487/RFC8794, July 2020, <https://www.rfc-editor.org/info/rfc8794>.

29. Informative References

[MCF] "Media Container Format", 17 July 2002, <http://mukoli.free.fr/mcf/mcf.html>.

Authors' Addresses

Steve Lhomme

Email: slhomme@matroska.org

Moritz Bunkus

Email: moritz@bunkus.org

Dave Rice

Email: dave@dericed.com