mp4文件格式解析

smallest_one 关注

4 2018.08.26 16:59:43 字数 5,088 阅读 35,701

目录

1. 概述
2. mp4文件基本信息
3. 封装格式重要概念
4. 重要box介绍
5. 其他box介绍
6. 实用技术
7. 开源软件

参考

- [1] ISO/IEC 14496-12:2015
- [2] wikipedia/MPEG-4
- [3] wikipedia/ISO base media file format
- [4] wikipedia/MPEG-4 Part 14
- [5] Tocy/多媒体文件格式之MP4
- [6] Phil Cluff/What's in the box_!
- [7] github.com/gpac/mp4box.js
- [8] tao/Android音视频系列：视频容器操作篇 -- mp4容器打包实现
- [9] LiaoJunXiong/使用gpac封装mp4
- [10] Jameson Steiner/Fun with Container Formats – Part 2

1. 概述

mp4或称MPEG-4 Part 14，是一种多媒体容器格式，扩展名为`.mp4`。

历史[6]：

- 2001年，apple的QuickTime格式，`.qt`和`.mov`的后缀名。
- 2001年，MPEG-4 Part1，把基于QuickTime的box布局的容器格式添加到了MPEG-4标准。
- 2004年，标准文档把编码和容器格式的说明分开。
  - MPEG-4 Part12，定义了容器格式通用的box结构，即ISO媒体文件格式(ISO base media file format, ISOBMFF)。
  - MPEG-4 Part14，基于Part12进行了细化，定义了用于存储MPEG-4内容的容器格式，即`.mp4`格式。

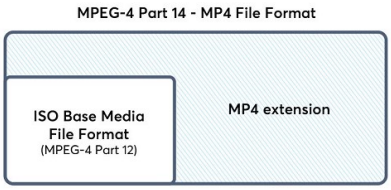


图1 MP4 File Format.png

以下是各标准文档的链接：

- QuickTime：QuickTime/QTFF
- MPEG-4 Part12：ISO/IEC 14496-12:2015
- MPEG-4 Part14：ISO/IEC 14496-14:2018，官网上这部分是付费的。

MP4文件由许多box组成，每个box包含不同的信息，这些box以树形结构的方式组织。以下是主要box的简要说明：

box 类型	说明
ftyp	file type，文件类型
moov	metadata container，存放媒体信息
mvhd	movie header，文件的总体信息，如时长，创建时间等
trak	track container，存放音频、视频流信息的容器
tkhd	track header，track的总体信息，如时长，宽高等
mdia	track media information container
mdhd	media header，存放TimeScale，trak需要TimeScale换算真实时间
hdlr	handler，指定trak类型是video/audio/hint
minf	media information container
stbl	sample table box 包含样本序号/时间//文件位置映射的信息
stsd	sample descriptions
stts	decoding time to sample，DTS-sample序号的映射表
ctts	composition time to sample CTS(创作时间)-DTS对应sample序号的映射表
stsc	sample-to-chunk，sample和chunk的映射表
stsz/stz2	sample size，每个sample的大小
stss	sync sample table，关键帧列表
stco/co64	chunk offset，每个chunk的文件偏移
mdat	media data container，具体的媒体数据

图2 主要box说明

根节点之下，主要包含三个节点：ftyp、moov、mdat。

- ftyp：文件类型。描述遵从的规范的版本。
- moov box：媒体的metadata信息。
- mdat：具体的媒体数据。

说明：在 mp4 中默认写入字节序是 Big-Endian的。

2. mp4文件基本信息

分析mp4文件的工具：

1. mp4box.js：一个在线解析mp4的工具。
2. bento4：包含mp4dump、mp4edit、mp4encrypt等工具。

3. MP4Box：类似于bento4，包含很全面的工具。
4. mp4info.exe: windows平台图形界面展示mp4基本信息的工具。

下图为使用mp4info.exe打开mp4文件的界面：

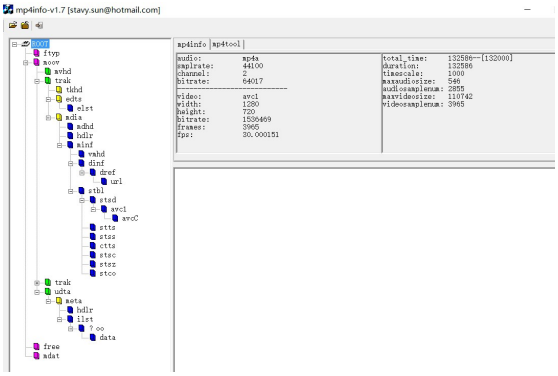


图3 mp4info.PNG

mp4文件基本信息
audio信息：

1. smplrate：sample rate(采样率)。
2. channel：通道个数。
3. bitrate：比特率。
4. audiosamplenum：音频sample的个数。

video信息：

1. width、height：视频的宽/高。
2. bitrate：比特率(码率)，秒为单位。等于视频总的大小/时长。
3. frames：视频帧数。
4. fps：帧率(frame per second)。
5. total_time：时间长度，ms为单位。等于duration/timescale。
6. timescale：时间的粒度，1000表示1000个单位为1s。
7. duration：时间粒度的个数。
8. videosamplenum：视频sample的个数。

3. 封装格式重要概念

3.1 box

mp4文件由若干个box组成。下面是box结构的一个示意图。

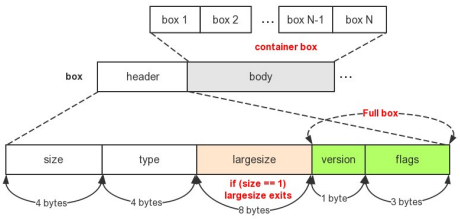


图4 mp4_box.png

- box由header和body组成，header指明box的size和type。size是包含box header的整个box的大小。
- box type，通常是4个ASCII码的字符如“ftyp”、“moov”等，这些box type都是已经预定义好的，表示固定的含义。如果是“uuid”，表示该box为用户自定义扩展类型，如果box type是未定义的，应该将其忽略。
- 如果header中的size为1，则表示box长度需要更多的bits位来描述，在后面会有一个64bits位的largesize用来描述box的长度。如果size为0，表示该box为文件的最后一个box，文件结尾（同样只存在于“mdat”类型的box中）。
- box中可以包含box，这种box称为container box。
- box分为两种，Box和Fullbox。FullBox 是 Box 的扩展，Header 中增加了version 和 flags字段，分别定义如下：

```
aligned(8) class Box (unsigned int(32) boxtype,
    optional unsigned int(8)[16] extended_type) {
    unsigned int(32) size;
    unsigned int(32) type = boxtype;
    if (size==1) {
        unsigned int(64) largesize;
    } else if (size==0) {
    // box extends to end of file
    }
    if (boxtype==‘uuid’) {
        unsigned int(8)[16] usertype = extended_type;
    }
}
```

FullBox有version和flags字段，

```
aligned(8) class FullBox(unsigned int(32) boxtype, unsigned int(8) v, bit(24) f)
extends Box(boxtype) {
    unsigned int(8) version = v;
    bit(24) flags = f;
}
```

3.2 track

一些sample的集合，对于媒体数据来说，track表示一个视频或者音频序列。

3.3 sample

video sample即为一帧或者一组连续视频帧，audio sample即为一段连续的音频。

3.4. sample table

指明sample时序和物理布局的表。

3.5. chunk

一个track的几个sample组成的单元。

mp4文件中，媒体内容在moov的box中。一个moov包含多个track。每个track就是一个随时间变化的媒体序列，track里的每个时间单位是一个sample，sample按照时间顺序排列。注意，一帧音频可以分解成多个音频sample，所以音频一般用sample作为单位，而不用帧。

4. 重要box介绍

4.1 Sample Table Box(stbl)

"stbl"是mp4文件中最复杂的一个box了，也是解开mp4文件格式的主干。
stbl：sample table是一个container box。
语法：

```
class SampleTableBox extends Box('stbl') {
}
```

其子box包括：

1. stsd：sample description box，样本的描述信息。
2. stts：time to sample box，sample解码时间的压缩表。
3. ctts：composition time to sample box，sample的CTS与DTS的时间差的压缩表。
4. stss：sync sample box，针对视频，关键帧的序号。
5. stsz/stz2：sample size box，每个sample的字节大小。
6. stsc：sample to chunk box，sample-chunk映射表。
7. stco/co64：chunk offset box，chunk在文件中的偏移。

sample是媒体数据存储的单位，存储在media的chunk中，chunk和sample的长度均可互不相同，如下图所示。

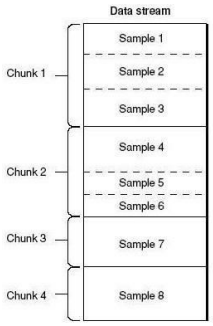


图5 sample.jpg

4.2 Sample Description Box(stsd)

存储了编码类型和初始化解码器需要的信息。
有与特定的track-type相关的信息，相同的track-type也会存在不同信息的情况如使用不一样的编码标准。
语法：

```
class SampleDescriptionBox (unsigned int(32) handler_type)
    extends FullBox('stsd', 0, 0){
        int i ;
        unsigned int(32) entry_count;
        for (i = 1 ; i <= entry_count ; i++){
            switch (handler_type){
                case 'soun': // for audio tracks
                AudioSampleEntry();
                break;
                case 'vide': // for video tracks
                VisualSampleEntry();
                break;
                case 'hint': // Hint track
                HintSampleEntry();
                break;
                case 'meta': // Metadata track
                MetadataSampleEntry();
                break;
            }
        }
    }
}
```

主要字段说明：

- entry count：entry的个数。
- handler_type：类型信息如"vide"、"soud"等，不同类型会提供不同的信息。

对于audio track，使用"AudioSampleEntry"类型信息。

```
abstract class SampleEntry (unsigned int(32) format)
    extends Box(format){
    const unsigned int(8)[6] reserved = 0;
    unsigned int(16) data_reference_index;
}

class AudioSampleEntry(codingname) extends SampleEntry (codingname){
    const unsigned int(32)[2] reserved = 0;
    template unsigned int(16) channelcount = 2;
```

```
    template unsigned int(16) samplesize = 16;
    unsigned int(16) pre_defined = 0;
    const unsigned int(16) reserved = 0 ;
    template unsigned int(32) samplerate = { default samplerate of media}<<16;
}
```

- format：视频或者音频的编码格式。
  - 比如aac音频的format值为mp4a。
  - AVC-1/H.264视频的format值为avc1。
- data_reference_index：利用这个索引可以检索与当前sample description关联的数据。数据引用存储在data reference box。
- channelcount：通道个数。
- samplesize：采样位数。默认是16比特。
- samplerate：采样率。[16.16]格式的数据。

对于video track，使用"VisualSampleEntry"类型信息。

```
class VisualSampleEntry(codingname) extends SampleEntry (codingname){
    unsigned int(16) pre_defined = 0;
    const unsigned int(16) reserved = 0;
    unsigned int(32)[3] pre_defined = 0;
    unsigned int(16) width;
    unsigned int(16) height;
    template unsigned int(32) horizresolution = 0x00480000; // 72 dpi
    template unsigned int(32) vertresolution = 0x00480000; // 72 dpi
    const unsigned int(32) reserved = 0;
    template unsigned int(16) frame_count = 1;
    string[32] compressorname;
    template unsigned int(16) depth = 0x0018;
    int(16) pre_defined = -1;
    // other boxes from derived specifications
    CleanApertureBox clap; // optional
    PixelAspectRatioBox pasp; // optional
}
```

- width、height：像素宽高。
- horizresolution、vertresolution：每英寸的像素值(dpi)，[16.16]格式的数据。
- frame_count：每个sample中的视频帧数，默认是1。可以是一个sample中有多帧数据。

4.3 Decoding Time to Sample Box（stts）

包含了一个压缩版本的表，通过这个表可以从解码时间映射到sample序号。表中的每一项是连续相同的编码时间增量(Decode Delta)的个数和编码时间增量。通过把时间增量累加就可以建立一个完整的time to sample表。
以下是Decoding Timing和Decode delta关系的一个图示：

Table 2 — Closed GOP Example

GOP	/--	---	---	---	---	---	--\	/--	---	---	---	---	---	--\
	I1	P4	B2	B3	P7	B5	B6	I8	P11	B9	B10	P14	B12	B13
DT	0	10	20	30	40	50	60	70	80	90	100	110	120	130
CT	10	40	20	30	70	50	60	80	110	90	100	140	120	130
Decode delta	10	10	10	10	10	10	10	10	10	10	10	10	10	10
Composition offset	10	30	0	0	30	0	0	10	30	0	0	30	0	0

图6 STTS_sample.PNG

- DT(decoding time)：编码时间。
- CT(composition time)：创作时间。
- Decode Delta：编码时间增量。
- Composition offset：显示时间同解码时间的差值，等于CT - DT。
  计算方式：

```
DT(n+1) = DT(n) + STTS(n) where STTS(n)
注:这里的STTS(n)是未压缩的Decode Delta表。

DT(i) = SUM(for j=0 to i-1 of delta(j))
```

语法：

```
class TimeToSampleBox
    extends FullBox('stts', version = 0, 0) {
    unsigned int(32) entry_count;
    int i;
    for (i=0; i < entry_count; i++) {
    unsigned int(32) sample_count;
    unsigned int(32) sample_delta;
    }
}
```

重要字段说明：

- entry_count：表中条目的个数。
- sample_count: 连续相同时间长度的sample个数。
- sample_delta：以timescale为单位的时间长度。

4.4 composition time to sample box（ctts）

这个box提供了decoding time到composition time的offset的表，用于计算pts。

- 这个表在Decoding time和composition time不一样的情况下时必须的。
- 如果box的version等于0，decoding time必须小于等于composition time，因而差值用一个无符号的数字表示。
- 有以下公式：

注：CTTS(n)是未压缩的表的第n个sample对应的offset。

语法：

```
class CompositionOffsetBox
```

```
    extends FullBox('ctts', version = 0, 0) {
    unsigned int(32) entry_count;
    int i;
    if (version==0) {
        for (i=0; i < entry_count; i++) {
            unsigned int(32) sample_count;
            unsigned int(32) sample_offset;
        }
    }
    else if (version == 1) {
        for (i=0; i < entry_count; i++) {
            unsigned int(32) sample_count;
            signed int(32) sample_offset;
        }
    }
}
```

主要字段说明：

- sample_count：连续相同的offset的个数。
- sample_offset：CT和DT之间的offset。

4.5 sync sample box(stss)

它包含media中的关键帧的sample表。如果此表不存在，说明每一个sample都是一个关键帧。
语法：

```
class SyncSampleBox
    extends FullBox('stss', version = 0, 0) {
    unsigned int(32) entry_count;
    int i;
    for (i=0; i < entry_count; i++) {
        unsigned int(32) sample_number;
    }
}
```

主要字段说明：

- sample_number：媒体流中同步sample的序号。

4.6 Sample Size Box（stsz/stz2）

包含sample的数量和每个sample的字节大小，这个box相对来说体积比较大的。
语法：

```
class SampleSizeBox extends FullBox('stsz', version = 0, 0) {
    unsigned int(32) sample_size;
    unsigned int(32) sample_count;
    if (sample_size==0) {
        for (i=1; i <= sample_count; i++) {
            unsigned int(32) entry_size;
        }
    }
}
```

主要字段说明：

- sample_size：指定默认的sample字节大小，如果所有sample的大小不一样，这个字段为0。
- sample_count：track中sample的数量。
- entry_size：每个sample的字节大小。

Compact Sample SizeBox(stz2)：一种压缩的sample大小存储方式。

```
class CompactSampleSizeBox extends FullBox('stz2', version = 0, 0) {
 unsigned int(24) reserved = 0;
 unisgned int(8) field_size;
 unsigned int(32) sample_count;
 for (i=1; i <= sample_count; i++) {
 unsigned int(field_size) entry_size;
 }
}
```

- field_size：指定表中条目的比特大小，取值为4、8或16。
  - (1) 如果使用值4，则每个字节存储两个sample的大小：
    entry[i] << 4 + entry[i + 1]。
  - (2) 如果大小没有填充满整数个字节，则用最后一个字节未使用部分填充0。
- sample_count是一个整数，它给出了下表中的条目数。
- entry_size是一个整数，它指定一个sample的大小，并根据其编号进行索引。

4.7 Sample To Chunk Box（stsc）

media中的sample被分为组成chunk。chunk可以有不同的大小，chunk内的sample可以有不同的大小。

通过stsc中的sample-chunk映射表可以找到包含指定sample的chunk，从而找到这个sample。结构相同的chunk可以聚集在一起形成一个entry，这个entry就是stsc映射表的表项。
语法：

```
class SampleToChunkBox
    extends FullBox('stsc', version = 0, 0) {
    unsigned int(32) entry_count;
    for (i=1; i <= entry_count; i++) {
        unsigned int(32) first_chunk;
        unsigned int(32) samples_per_chunk;
        unsigned int(32) sample_description_index;
    }
}
```

主要字段说明：

- first_chunk：一组chunk的第一个chunk的序号。
  - chunk的编号从1开始。
- samples_per_chunk：每个chunk有多少个sample。
- sample_desc_idx：stsd 中sample desc信息的索引。

把一组相同结构的chunk放在一起进行管理，是为了压缩文件大小。

用mp4box.js查看的stsc box的信息如下：

image.png

- 第一组chunk的first_chunk序号为1，每个chunk的sample个数为1，因为第二组chunk的first_chunk序号为2，可知第一组chunk中只有一个chunk。
- 第二组chunk的first_chunk序号为2，每个chunk的sample个数为2，因为第三组chunk的first_chunk序号为24，可知第二组chunk中有22个chunk，有44个sample。

4.8 Chunk Offset Box（stco/co64）

Chunk Offset表存储了每个chunk在文件中的位置，这样就可以直接在文件中找到媒体数据，而不用解析box。

- 需要注意的是一旦前面的box有了任何改变，这张表都要重新建立。

语法：

```
class ChunkOffsetBox
 extends FullBox('stco', version = 0, 0) {
 unsigned int(32) entry_count;
 for (i=1; i <= entry_count; i++) {
 unsigned int(32) chunk_offset;
 }
}
```

主要字段说明：

- chunk_offset：chunk在文件中的位置。

stco 有两种形式，如果你的视频过大的话，就有可能造成 chunkoffset 超过 32bit 的限制。所以，这里针对大 Video 额外创建了一个 co64 的 Box。它的功效等价于 stco，也是用来表示 sample 在 mdat box 中的位置。只是，里面 chunk_offset 是 64bit 的。

```
aligned(8) class ChunkLargeOffsetBox extends FullBox('co64', version = 0, 0) {
    unsigned int(32) entry_count;
    for (i=1; i <= entry_count; i++) {
        unsigned int(64) chunk_offset;
    }
}
```

5. 其他box介绍

以下是ISO/IEC 14496-12:2015文档给出的box的描述图：

box.png

5.1 File Type Box(ftyp)

File Type Box一般在文件的开头，用来指示该 mp4文件使用的标准规范。为了早期规范版本兼容，允许不包含ftyp box。
语法：

```
class FileTypeBox
    extends Box('ftyp') {
    unsigned int(32) major_brand;// is a brand identifier
    unsigned int(32) minor_version;// is an informative integer for the minor version of
    unsigned int(32) compatible_brands[]; //is a list, to the end of the box, of brands
}
```

没有ftyp box的文件应该处理成ftyp的major_brand为'mp41'，minor_version为0，compatible_brands只包含一个'mp41'。

5.2 Movie Box(moov)

Movie Box包含了文件媒体的metadata信息，"moov"是一个container box，具体内容信息在其子box中。一般情况下，"moov"会紧随着"ftyp"。

"moov"中包含1个"mvhd"和若干个"trak"。其中"mvhd"是header box，一般作为"moov"的第一个子box出现。"trak"包含了一个track的相关信息，是一个container box。
语法：

```
class MovieBox extends Box('moov'){
}
```

5.3 Movie Header Box（mvhd）

语法：

```
class MovieHeaderBox extends FullBox('mvhd', version, 0) {
    if (version==1) {
    unsigned int(64) creation_time;
    unsigned int(64) modification_time;
    unsigned int(32) timescale;
    unsigned int(64) duration;
    } else { // version==0
    unsigned int(32) creation_time;
    unsigned int(32) modification_time;
    unsigned int(32) timescale;
    unsigned int(32) duration;
    }
    template int(32) rate = 0x00010000; // typically 1.0
    template int(16) volume = 0x0100; // typically, full volume
    const bit(16) reserved = 0;
    const unsigned int(32)[2] reserved = 0;
    template int(32)[9] matrix =
    { 0x00010000,0,0,0,0x00010000,0,0,0,0x40000000 };
    // Unity matrix
    bit(32)[6] pre_defined = 0;
    unsigned int(32) next_track_ID;
}
```

主要字段含义：

- version：box版本，0或1，一般为0。
- creation time：创建时间（相对于UTC时间1904-01-01零点的秒数）。
- modification time：修改时间 。
- timescale：文件媒体在1秒时间内的刻度值，可理解为1秒长度的时间单元数。
- duration：该track的时间长度，用duration和time scale值可以计算track时长，比如audio track的time scale = 8000, duration = 560128，时长为70.016，video track的time scale = 600, duration = 42000，时长为70。
- rate：推荐播放速率，高16位和低16位分别为小数点整数部分和小数部分，即[16.16] 格式，该值为1.0（0x00010000）表示正常前向播放。
- volume：推荐播放音量，[8.8] 格式，1.0（0x0100）表示最大音量。

5.4 Track Box（trak）

Track Box是一个container box，其子box包含了该track的媒体数据引用和描述（hint track除外）。一个mp4文件可以包含多个track，且至少有一个track，track之间是独立，有自己的时间和空间信息。“trak”必须包含一个“tkhd”和一个“mdia”，此外还有很多可选的box。其中“tkhd”为track header box，“mdia”为media box，该box是一个包含一些track媒体数据信息box的container box。
语法：

```
class TrackBox extends Box('trak') {
}
```

5.5 Track Header Box（tkhd）

语法：

```
class TrackHeaderBox
    extends FullBox('tkhd', version, flags){
    if (version==1) {
    unsigned int(64) creation_time;
    unsigned int(64) modification_time;
    unsigned int(32) track_ID;
    const unsigned int(32) reserved = 0;
    unsigned int(64) duration;
    } else { // version==0
    unsigned int(32) creation_time;
    unsigned int(32) modification_time;
    unsigned int(32) track_ID;
    const unsigned int(32) reserved = 0;
    unsigned int(32) duration;
    }
    const unsigned int(32)[2] reserved = 0;
    template int(16) layer = 0;
    template int(16) alternate_group = 0;
    template int(16) volume = {if track_is_audio 0x0100 else 0};
    const unsigned int(16) reserved = 0;
    template int(32)[9] matrix=
    { 0x00010000,0,0,0,0x00010000,0,0,0,0x40000000 };
    // unity matrix
    unsigned int(32) width;
    unsigned int(32) height;
}
```

主要字段含义：

- version：box版本，0或1，一般为0。
- flags：24-bit整数，按位或操作结果值，预定义的值(0x000001，track_enabled，表示track是有效的)、(0x000002，track_in_movie，表示该track在播放中被使用)、(0x000004，track_in_preview，表示track在预览时被使用。
- track id：track id号，不能重复且不能为0。
- duration：track的时间长度，计量单位timescale在mvhd中。
- volume：[8.8] 格式，如果为音频track，1.0（0x0100）表示最大音量；否则为0。
- width：宽，[16.16] 格式值。
- height：高，[16.16] 格式值，不必与sample的像素尺寸一致，用于播放时的展示宽高。

5.6 Media Box（mdia）

Media Box也是个container box。
语法：

```
class MediaBox extends Box('mdia') {
}
```

其子box的结构和种类还是比较复杂的。
“mdia”定义了track媒体类型以及sample数据，描述sample信息。

一个“mdia”必须包含如下容器：

- 一个Media Header Atom(mdhd)
- 一个Handler Reference(hdlr)
- 一个media information(minf)和User Data

下面依次看一下这几个box的结构。

5.7 Media Header Box（mdhd）

mdhd 和 tkhd，内容大致都是一样的。不过tkhd 通常是对指定的 track 设定相关属性和内容。而 mdhd 是针对于独立的 media 来设置的。不过两者一般都是一样的。
语法：

```
class MediaHeaderBox extends FullBox('mdhd', version, 0) {
    if (version==1) {
    unsigned int(64) creation_time;
    unsigned int(64) modification_time;
    unsigned int(32) timescale;
    unsigned int(64) duration;
    } else { // version==0
    unsigned int(32) creation_time;
    unsigned int(32) modification_time;
    unsigned int(32) timescale;
    unsigned int(32) duration;
    }
    bit(1) pad = 0;
    unsigned int(5)[3] language; // ISO-639-2/T language code
    unsigned int(16) pre_defined = 0;
}
```

主要字段含义：

- version: box版本，0或1，一般为0。
- timescale: 同mvhd中的timescale。
- duration: track的时间长度。
- language: 媒体语言码。最高位为0，后面15位为3个字符（见ISO 639-2/T标准中定义）。

5.8 Handler Reference Box（hdlr）

“hdlr”解释了媒体的播放过程信息，该box也可以被包含在meta box（meta）中。
语法：

```
class HandlerBox extends FullBox('hdlr', version = 0, 0) {
    unsigned int(32) pre_defined = 0;
    unsigned int(32) handler_type;
    const unsigned int(32)[3] reserved = 0;
    string name;
}
```

主要字段含义：

- handler type: 在media box中，该值为4个字符，会有以下取值：

```
'vide' Video track
'soun' Audio track
'hint' Hint track
'meta' Timed Metadata track
'auxv' Auxiliary Video track
```

- name: human-readable name for the track
  type，以'\0'结尾的 UTF-8 字符串。用于调试后者检查的目的。

5.9 Media Information Box（minf）

重要的容器 box，"minf"存储了解释track媒体数据的handler-specific信息，media handler用这些信息将媒体时间映射到媒体数据并进行处理。"minf"是一个container box，其实际内容由子box说明。
语法：

```
class MediaInformationBox extends Box('minf') {
}
```

"minf"中的信息格式和内容与媒体类型以及解释媒体数据的media handler密切相关，其他media handler不知道如何解释这些信息。

一般情况下，"minf"包含一个header box，一个"dinf"和一个"stbl"，其中，header box根据track type（即media handler type）分为"vmhd"、"smhd"、"hmhd"和"nmhd"，"dinf"为data information box，"stbl"为sample table box。下面分别介绍。

5.10 Media Information Header Box（vmhd、smhd、hmhd、nmhd）

vmhd、smhd这两个box在解析时，非不可或缺的（有时候得看播放器），缺了的话，有可能会被认为格式不正确。

Video Media Header Box（vmhd）
语法：

```
class VideoMediaHeaderBox
    extends FullBox('vmhd', version = 0, 1) {
    template unsigned int(16) graphicsmode = 0; // copy, see below
    template unsigned int(16)[3] opcolor = {0, 0, 0};
}
```

主要字段含义：

- graphics mode：视频合成模式，为0时拷贝原始图像，否则与opcolor进行合成。
- opcolor: 一组(red，green，blue)，graphics modes使用。

Sound Media Header Box（smhd）
语法：

```
class SoundMediaHeaderBox
    extends FullBox('smhd', version = 0, 0) {
    template int(16) balance = 0;
    const unsigned int(16) reserved = 0;
}
```

主要字段含义：

- balance：立体声平衡，[8.8] 格式值，一般为0表示中间，-1.0表示全部左声道，1.0表示全部右声道。

Hint Media Header Box（hmhd）
Null Media Header Box（nmhd）
非视音频媒体使用该box。

5.11 Data Information Box（dinf）

"dinf"解释如何定位媒体信息，是一个container box。
语法：

```
class DataInformationBox extends Box('dinf') {
}
```

"dinf"一般包含一个"dref"(data reference box)。
"dref"下会包含若干个"url"或"urn"，这些box组成一个表，用来定位track数据。简单的说，track可以被分成若干段，每一段都可以根据"url"或"urn"指向的地址来获取数据，sample描述中会用这些片段的序号将这些片段组成一个完整的track。一般情况下，当数据被完全包含在文件中时，"url"或"urn"中的定位字符串是空的。

"dref"的语法：

```
class DataEntryUrlBox (bit(24) flags)
    extends FullBox('url ', version = 0, flags) {
    string location;
}
class DataEntryUrnBox (bit(24) flags)
    extends FullBox('urn ', version = 0, flags) {
    string name;
    string location;
}
class DataReferenceBox
    extends FullBox('dref', version = 0, 0) {
    unsigned int(32) entry_count;
    for (i=1; i <= entry_count; i++) {
    DataEntryBox(entry_version, entry_flags) data_entry;
```

```
        }
    }
```

主要字段含义：

- entry count：“url”或“urn”表的元素个数。
- entry_version：entry格式的版本。
- entry_flags：当“url”或“urn”的box flag为1时，表示数据在该文件的Moov Box中。
- “url”或“urn”都是box，“url”的内容为location字符串，“urn“的内容为名称字符串和location字符串。

6. 实用技术

6.1 moov移到mdat前面

ffmpeg默认情况下生成moov是在mdat写完成之后再写入，所以moov是在mdat的后面，使用faststart参数可以将moov移到mdat前面。

```
./ffmpeg -i demo.flv -c copy output.mp4
```

- moov在mdat后面

image.png

使用faststart参数

```
ffmpeg -i out.flv -c copy -movflags faststart out2.mp4
```

- moov在mdat前面

image.png

6.2 如何实现seek

stbl.png

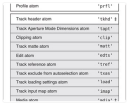
例如，我们需要seek到30s。
需要做如下工作：

1. 使用timescale将目标时间标准化。timescale为90000，30*90000=2700000。
2. 通过time-to-sample box找到指定track的给定时间之前的第一个sample number。2700000/3000 = 900。

3. 通过sync sample table查询sample number之前的第一个sync sample。对应为795的sample。
4. 通过sample-to-chunk table查找到对应的chunk number。
   对应的chunk号假设是400。
5. 通过chunk offset box查找到对应chunk在文件中的起始偏移量。第14个chunk的offset是3481072。
6. 最后使用sample-to-chunk box和sample size box的信息计算出该chunk中需要读取的sample在文件中的起始偏移量，即完成seek。

7. 开源软件

- gpac：开源的多媒体工具包，包括用于MP4打包的mp4box等。
- mp4v2：提供了API来创建和修改mp4文件。



smallest_one

关注

总资产6

利用nginx-rtmp搭建视频点播、直播、HLS服务器

阅读 2,320

x264 设置日志级别

阅读 684

推荐阅读

实操|rrBLUP包RRBLUP

阅读 227

c语言 关于0长度数组的使用

阅读 66

base64与File转换

阅读 125

安卓音频开发(四)使用lame把wav转mp3

阅读 206

RNAseq-踩坑02 -- Aligment 比对率

写下你的评论...

评论3

赞47