

DoubleLi
qq: 517712484 wx: ldbgliet

博客园 :: 首页 :: 博文 :: 闪存 :: 新随笔 :: 联系 :: 订阅 [RSS](#) :: 管理 :3867 随笔 :: 2 文章 :: 473 评论 :: 1286万 阅读

2021年10月							
日	一	二	三	四	五	六	
26	27	28	29	30	1	2	
3	4	5	6	7	8	9	
10	11	12	13	14	15	16	
17	18	19	20	21	22	23	
24	25	26	27	28	29	30	
31	1	2	3	4	5	6	

公告

昵称: DoubleLi
园龄: 11年9个月
粉丝: 2080
关注: 29
+加关注

搜索

- 常用链接
- [我的随笔](#)
[我的评论](#)
[我的参与](#)
[最新评论](#)
[我的标签](#)

- 随笔分类 (5164)
- [android\(2\)](#)
[ASP.NET\(30\)](#)
[ASP.NET MVC\(11\)](#)
[Boost\(118\)](#)
[c#\(10\)](#)
[C++/C\(778\)](#)
[c++11\(15\)](#)
[cmake/autotool\(66\)](#)
[com/ATL/Activex\(75\)](#)
[Css\(16\)](#)
[CImage\(12\)](#)
[darwin stream server\(3\)](#)
[DataBase\(32\)](#)
[DirectX\(16\)](#)
[Extjs\(13\)](#)
[更多](#)

- 随笔档案 (3864)
- [2021年10月\(33\)](#)
[2021年9月\(4\)](#)
[2021年8月\(10\)](#)
[2021年7月\(43\)](#)
[2021年6月\(1\)](#)
[2021年5月\(29\)](#)
[2021年4月\(15\)](#)
[2021年3月\(13\)](#)
[2021年2月\(96\)](#)
[2021年1月\(47\)](#)
[2020年12月\(2\)](#)
[2020年11月\(27\)](#)
[2020年10月\(44\)](#)
[2020年9月\(14\)](#)
[2020年8月\(4\)](#)
[更多](#)

- 文章分类 (2)
- [SilverLight\(1\)](#)
[sql server\(1\)](#)

MP4封装格式介绍 -- Atom结构

原文：<https://blog.csdn.net/uyy203/article/details/79283087>

视频封装格式是指视频数据如何存储的，视频编码格式是指原始视频数据如何编码为二进制数据码流。编码后的视频数据最终通过视频封装格式存储为视频文件。

本文对视频封装格式MP4做一些介绍。主要介绍了Atom结构和Atom中数据的意义。

一、MP4封装格式

ISO/IEC 14496-12(MPEG-4 Part 12 ISO base media file format)定义了一种通用的数字媒体文件格式标准，其基础是2001年版的Apple QuickTime文件格式。ISO/IEC 14496-14标准中定义了ISO/IEC 14496-12的一种多媒体容器格式实现，即我们现在熟知的MP4(MPEG-4 Part 14)，它属于MPEG-4的一部分。

MP4文件由若干称为Atom（或称为box）的数据对象组成，每个Atom的起首为四个字节的长度（Big Endian）和四个字节的类型标识，数据长度和类型标志都可以扩展。Atom可以嵌套，即其数据域可以由若干其它Atom组成，从而实现结构化的数据。

MP4是一种描述较为全面的容器格式，被认为可以在其中嵌入任何形式的数据，包括各种编码的视频、音频，我们常见的大部分的MP4文件存放的AVC(H.264)或MPEG-4(Part 2)编码的视频和AAC编码的音频。

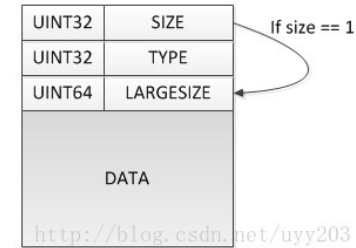
MP4格式的官方文件后缀名是“.mp4”，还有其他的以mp4为基础进行的扩展或者是缩水版本的格式，包括：M4V, 3GP, F4V等。

说明：
QuickTime影片格式是Apple公司开发的一种音频、视频文件格式，用于存储常用数字媒体类型。其扩展名为.mov。

二、MP4文件结构

MP4是由Atom嵌套来存放媒体信息。Atom的基本结构是：
[4bytes atom length] [4bytes atom name] [8bytes largesize, if size ==1] [contents of the atom, if any]

结构如图：



image

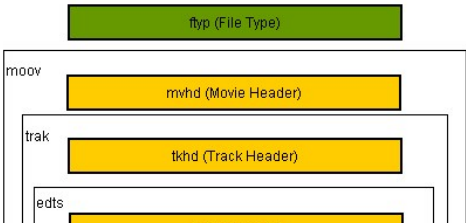
SIZE表示整个Atom所占用的大小，包括header部分。如果Atom很大，超过了uint32的最大数值(例如存放具体视频数据的mdat)，size就被设置为1，并用接下来的LARGESIZE来存放数据大小。

TYPE表示这个Atom的类型，主要有ftyp、moov、mdat等。

LARGESIZE，如果SIZE==1，则使用8bytes uint64来存储该Atom大小。

DATA是实际的数据。

MP4文件含有很多Atom，主要的Atom和嵌套结构如图。



参考博客

linux驱动
回忆未来-向东
Nginx模块开发与原理剖析
大坡3D软件开发
Dean Chen的专栏
Sloan
音视频FFmpeg等
opencv教程
个人开发历程知识库
关注DirectX
chenyujing1234
morewindows
雷霄骅(leixiaohua1020)的专栏
ffmpeg参考
webrtc参考—
更多

阅读排行榜

1. Nginx之location 匹配规则详解(240898)
2. cmake使用方法详解(178488)
3. MinGW安装和使用(103448)
4. RTMP、RTSP、HTTP视频协议详解 (附：直播流地址、播放软件) (102153)
5. C语言字符串操作总结大全(超详细)(94264)

评论排行榜

1. 非IE内核浏览器支持activex插件(37)
2. Nginx之location 匹配规则详解(19)
3. Javascript中定义类(15)
4. C++中的头文件和源文件(9)
5. RTSP协议详解(8)

推荐排行榜

1. C++中的头文件和源文件(25)
2. Nginx之location 匹配规则详解(22)
3. Javascript中定义类(12)
4. JavaScript中typeof知多少？(11)
5. MinGW安装和使用(9)

最新评论

1. Re:windows下搭建nginx-rtmp服务器
configuration-nginx.bat执行报错啊 'auto' 不是内部或外部命令，也不是可运行的程序 或批处理文件。
'--conf-path' 不是内部或外部命令，也不是可运行的程序 或批...
--猫爷°
2. Re:深入理解linux系统下proc文件系统内容

怎么联系作者

--o=

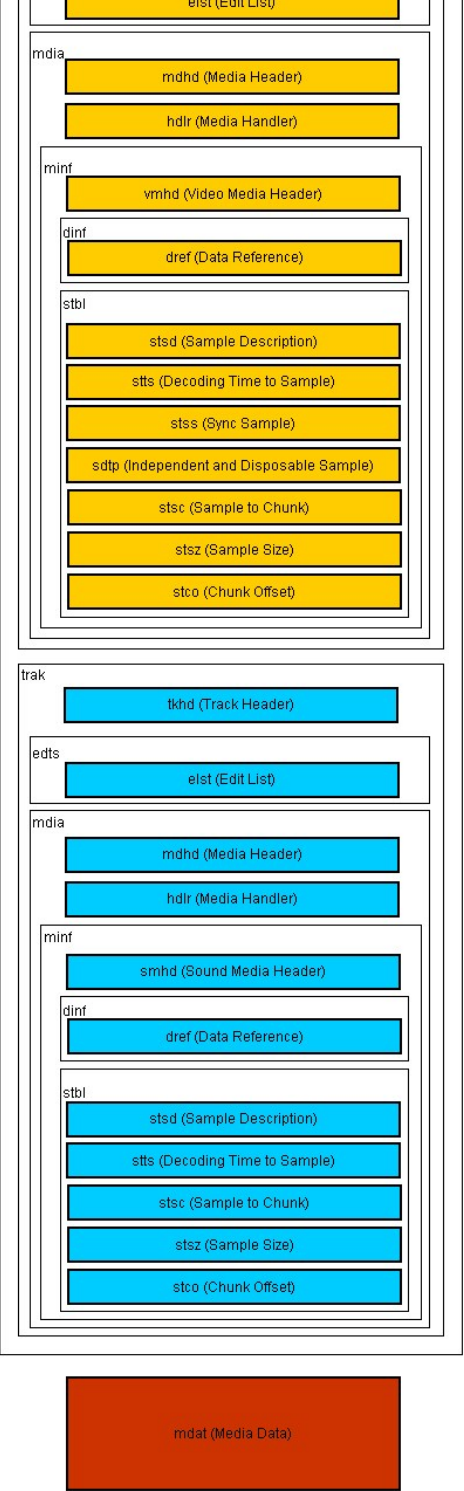
3. Re:go mod模式下引用本地包/模块(module)的方法

go mod用法，不错

--立志做一个好的程序员

4. Re:谷歌浏览器Chrome播放rtsp视频流解决方案
目前市面上已经有很成熟且商用Chrome播放海康威视大华的H.264或H.265的RTSP视频流解决方案了，就是猿大师中间件，底层调用VLC的ActiveX控件可实现网页中内嵌播放多路RTSP的实时...
--喵大侠

5. Re:如何使用UDP进行跨网段广播
主机A：192.168.3.100 子网掩码255.255.0.0 (手动临时修改) 主机



三、一个典型的MP4文件实例

moov保存了视频的基本信息，mdat保存视频和音频数据。这两个Atom顺序不固定。

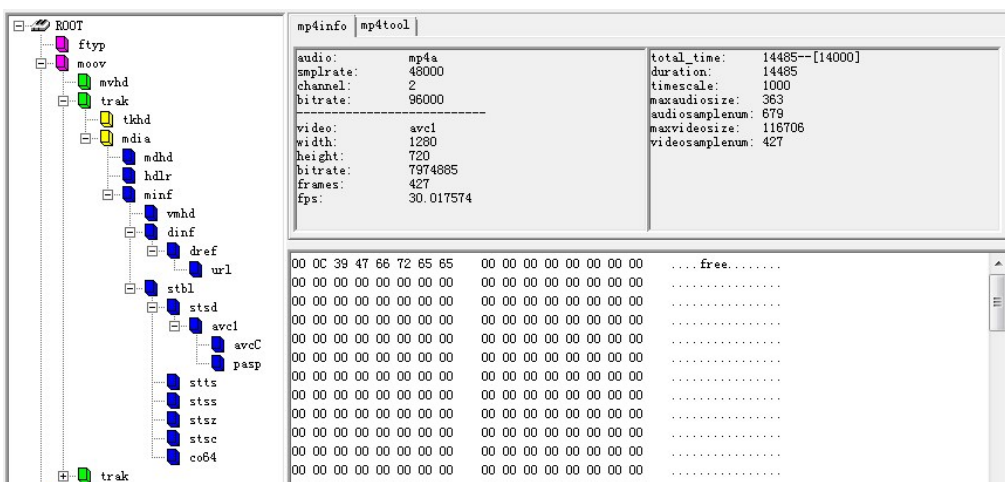
一个moov在mdat之后的MP4文件的Atom结构：

image

由于moov中保存了视频数据的索引，对于在线播放的场景，需要moov在mdat之前，才能流式读取视频数据。

部分摄像设备生成的MP4文件中，moov在mdat之前，两者之间可能还存在一个Atom free，即moov-free-mdat。free中为全0，只是用于占位。

一个moov在mdat之前的MP4文件的Atom结构：





image

四、ATOM说明

1、FTYP

ftyp是整个文件的第一个Atom，通过判断该Atom来确定文件的类型。该Atom有且只有1个，并且只能被包含在文件层，而不能被其他Atom包含。该Atom应该被放在文件的最开始，指示文件的相关信息。

文件的最开始的四个字节就是“ftyp”Atom的大小，然后是该Atom的类型。“ftyp”的body依次包括1个32位的major brand（4个字符），1个32位的minor version（整数）和1个以32位（4个字符）为单位元素的数组compatible brands。这些都是用来指示文件应用级别的信息。以一个MP4文件的“ftyp”Atom为例，如下所示：

```
00000000: 0000 0018 6674 7970 6d70 3432 0000 0000 ....ftypmp42....
```

```
0000010: 6d70 3432 6d70 3431 0528 834f 6d64 6174 mp42mp41.(Omdat
```

其中，

- (1) 0x00 00 00 18是“ftyp”Atom的大小，为24个字节，这在一般情况下为一个固定值。
- (2) 0x66 74 79 70是“ftyp”四个字符的ASCII值，也就是该Atom的类型。
- (3) 0x6D 70 34 32是major brand，这里为“mp42”，对于不同的文件，该值可能是不一样的。
- (4) 0x00 00 00 00是minor version。
- (5) 0x6D 70 34 32和0x6D 70 34 31是compatible brands，“mp42”和“mp41”

FTYP到底是什么呢？

ftyp就是一个由四个字符组成的码字，用来标识编码类型、兼容性或者媒体文件的用途。它存在于MP4文件和MOV文件中，当然，也存在于3GP文件中。

虽然MP4文件、MOV文件和3GP文件采用了相同的封装标准，但是由于是由不同的厂商合成，因此还是存在差别的。即使是同一种媒体文件，比如MP4文件，由不同developers开发的MP4也是存在差别的。ftyp简单的说就是为了标识它的developer是谁，兼容哪些标准等。

比如上面的例子，“mp42”表示它的major brand是MP4 v2 [ISO 14496-14]，而“mp42”和“mp41”则表示它的compatible brands是MP4 v2 [ISO 14496-14]和MP4 v1 [ISO 14496-1:ch13]。

2、FREE

free是可选的，如果存在，则通常出现在moov与mdat之间，即moov-free-mdat。

free中的数据通常为全0，其作用相当于占位符，在实时拍摄视频，moov数据增多时分配给moov使用。

因为设备录制视频时并不能预先知道视频数据大小，如果moov在mdat之前，随着拍摄mdat的数据会增加，moov数据也会增多，如果没有free预留的空间，则要不不停的向后移动mdat数据以腾出moov空间。

3、MOOV

moov中主要保存了媒体的时间信息、trak信息和媒体索引等。

3.1 媒体时间信息

moov-mvhd中有一个time scale，以1/n秒的形式给出一个总的时间粒度，moov-trak-tkhd中以此时间粒度给出各个track的duration；

3.2 trak信息

moov中通常包含两个trak，一个视频索引，一个音频索引。

3.2.1 trak类型

trak的类型在moov-trak-mdia-hdlr中给出，包括‘vide’，‘soun’和‘hint’三种。

3.2.2 trak的时间

moov-trak-mdia-mdhd中以1/n秒的形式给出各个媒体的时间粒度以及以此时间粒度为单位的duration。

moov-trak-mdia-minf-stbl-stts中有媒体帧之间的时间间隔，单位是moov-trak-tkhd中的时间粒度。

3.2.3 索引信息

moov-trak-mdia-minf-stbl比较重要，其中保存了解码器需要的信息和索引信息，以下Atom都是stbl Atom的孩子。

stsd中保存了解码器需要的媒体描述信息。

stts {n: sample delta}

stsc {[1, 3], {3, 2}}

stsz {sz1, sz2, sz3, ... szn}





stco {co1, co2, co3, ... cok}

<http://blog.csdn.net/uyy203>

image

stss, Sync Sample Atom

标识了媒体流中的关键帧，提供了随机访问点。每个entry标识了一个关键帧。
关键帧号是按照增长顺序排列的。如果该Atom不存在，表示所有帧都是关键帧。

Number	Sample 1
Number	Sample 2
Number	Sample 3
Number	Sample 4
Number	Sample 5

image

Sync Sample Table的布局

stts, Time-To-Sample Atoms

stts给出每个数据帧之间的时间间隔，单位是moov-trak-tkhd中的时间粒度。
Atom的每个entry给出了具有相同时间间隔的连续帧的个数，这些帧的时间间隔值，结构如图。

Sample count	Sample duration	Field
4	4	Bytes

image

Time-To-Sample的table entry布局

如果连续的帧有相同的时长，他们会被放在同一个entry中。如果所有的帧具有相同的时长，那么Atom中就只有一个entry。

stts实例

下图通过3个entries来描述9个帧。需要说明的是，这里的entry和chunk不是对应的。例如，4、5、6帧可以在同一个chunk中，但是，由于它们的时长不同，4帧的时长为3，而5、6帧的时长为1，因此，保存在不同的entry中。

Sample count	Sample duration
4	3
2	1
3	2

image

stco/co64, Chunk Offset Atom

stco/co64给出每个数据Chunk在文件中的偏移。Chunk Offset Atom的每个entry给出了每个chunk在文件中的偏移。
如果Chunk Offset Atom的类型为stco，则保存的偏移量是32位；如果是co64，则保存的偏移量是64位的。布局如图。

Offset	Chunk 1
--------	---------

Offset	Chunk 2
Offset	Chunk 3
Offset	Chunk 4
Offset	Chunk 5

image

chunk offset table的布局

需要注意的是，该Atom中只给出了每个chunk的偏移量，并没有给出每个sample的偏移量。因此，如果要获得每个sample的偏移量，还需要用到Sample Size Table和Sample-To-Chunk Table。

stsc, Sample-To-Chunk Atom

stsc给出各个数据Chunk中包含的数据帧。一个chunk可能会包含一个或者几个帧。每个chunk会有不同的size，每个chunk中的帧也会有不同的size。entry中保存了第一个chunk号、每个chunk包含的帧数、帧描述ID。

First chunk	Samples per chunk	Sample description ID	Fields
4	4	4	Bytes

image

Sample-To-Chunk Atom的table entry布局

每个entry包含一组chunk，其中每个chunk的帧数相同。而且，这些chunk中的每个帧都必须使用相同的帧描述。

如果chunk中的帧数或者帧描述改变，必须创建一个新的entry。

如果所有的chunk包含的帧数和帧描述相同，那么只有一个entry。

stsc实例

图中表示至少有5个chunk，第1、2个chunk分别包含3个帧，帧描述ID是23；第3、4个chunk分别包含1个帧，帧描述ID是23；第5个及以后的chunk，包含1个帧，帧描述ID是24。

对于最后一个entry需要特殊的处理，因为无法判断什么时候结束。

First chunk	Samples per chunk	Sample description ID
1	3	23
3	1	23
5	1	24

image

stsz, Sample Size Atom

Size	Sample 1
Size	Sample 2
Size	Sample 3
Size	Sample 4
Size	Sample 5

image

sample size table的布局

3.3 用户定义数据

udta中保存了用户定义数据，例如iTune使用的meta数据就保存在udta中。

3.4 用户扩展数据

Atom的扩展通过uuid实现。用户可以使用类型为‘ uuid’ 的Atom，以16个特定的字节作为标识，定义自己的数据格式。

4、MDAT

所有媒体数据统一存放在mdat中，没有同步字，没有分隔符，只能根据索引进行访问。
mdat的位置比较灵活，可以位于moov之前，也可以位于moov之后，但必须和stbl中的信息保持一致。

另外，在写mp4文件的时候，对于mdat这个Atom，一般是先将Atom size填写0，待数据写完之后，再反过来填入具体大小。

5、其他ATOM

5.1、THMB

Atom moov @ 13840405 of size: 355190, ends @ 14195595
.....
Atom udta @ 14146330 of size: 49265, ends @ 14195595
Atom thmb @ 14146338 of size: 49197, ends @ 14195535
Atom cpri [zho] @ 14195535 of size: 30, ends @ 14195565
Atom perf [zho] @ 14195565 of size: 30, ends @ 14195595

thmb 结构：
atom size (49197)
atom type (thmb)
thmb size (0)
thmb type (jpeg)
thmb data

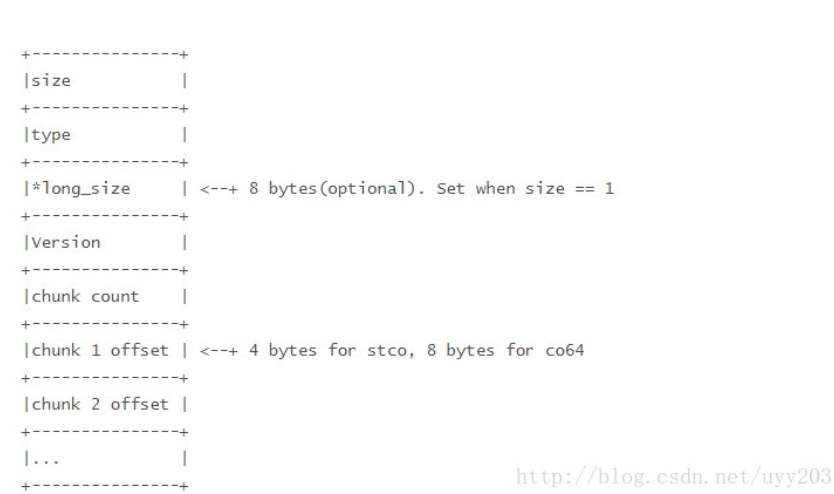
五、实际应用场景

1、把文件尾部的moov移到文件头

对于流媒体播放，如果mdat的位置在moov之前，通过流的方式播放文件会出现问题，因为没有办法在一开始就获得文件的媒体信息和索引。
这种情况需要对视频做预处理，把moov移动到文件头。通过遍历Atom可以很容易找到moov Atom。

需要注意的是，移动moov到文件头，同时需要修改moov中的stco/co64。因为这里保存了chunk数据的偏移量，移动moov后，需要根据moov的新位置更新stco/co64。

stco/co64的结构如下：



<http://blog.csdn.net/uyy203>

image

可参考 [python开源库 qtfaststart](#)。

2、分割MP4文件

在视频点播服务中，需要将MP4文件分割为多个分片，此时需要获取关键帧、切割时间轴、更新moov和生成各个分片文件。

可参考 <http://www.cnblogs.com/haibindev/archive/2011/10/17/2214518.html>

附 查看Atom相关工具

Windows：mp4info，解析和用树形显示文件的atom

附录: ISO/IEC 14496 MPEG的协议标准

ISO/IEC 14496是MPEG专家组制定的MPEG-4标准于1998年10月公布第1版, 1999年1月成为国际标准, 1999年12月公布了第2版, 2000年初成为国际标准。

全文分为27个部分:

- (1) ISO/IEC 14496-1系统部分, 描述视频和音频数据流的控制、同步以及混合方式(即混流 Multiplexing, 简称为MUX)。
- (2) ISO/IEC 14496-2视频部分, 定义了一个对各种视觉信息(包括自然视频、静止纹理、计算机合成图形等等)的编解码器。(例如XviD编码就属于MPEG-4 Part 2)。
- (3) ISO/IEC 14496-3音频部分, 定义了一个对各种音频信号进行编码的编解码器的集合。包括高级音频编码(Advanced Audio Coding, 缩写为AAC)的若干变形和其他一些音频/语音编码工具。
- (4) ISO/IEC 14496-4一致性部分, 定义了比特流和设备的一致性条件, 用来测试MPEG-4的实现。
- (5) ISO/IEC 14496-5参考软件, 提供了用于演示功能和说明本标准其他部分功能的软件。
- (6) ISO/IEC 14496-6多媒体传送整体框架DMIF, 这是MPEG-4应用层与传输网络的接口, 定义了通信协议, 使MPEG-4系统的数据流能进入各种传输网络。还包含一个存储格式MP4, 用于存储编码的场景。
- (7) ISO/IEC 14496-7优化的参考软件, 提供了对实现进行优化的例子(这里的实现指的是第五部分)。
- (8) ISO/IEC 14496-8在IP网络上传输, 定义了IP网络上传输MPEG-4内容的方式。
- (9) ISO/IEC 14496-9参考硬件描述, 提供了用于演示怎样在硬件上实现本标准其他部分功能的硬件设计方案
- (10) ISO/IEC 14496-10高级视频编码AVC, 定义了一个视频编解码器(codec)。AVC和XviD都属于MPEG-4编码, 但由于AVC属于MPEG-4 Part 10, 在技术特性上比属于MPEG-4 Part2的XviD要先进。另外, 它和ITU-T H.264标准是一致的, 故又称为H.264。
- (11) ISO/IEC 14496-11场景描述和应用引擎。
- (12) ISO/IEC 14496-12ISO媒体文件格式, 定义了一个存储媒体内容的文件格式。
- (13) ISO/IEC 14496-13知识产权管理和保护(IPMP: Intellectual Property Management and Protection)扩展。
- (14) ISO/IEC 14496-14MP4文件格式, 定义了基于第十二部分的用于存储MPEG-4内容的容器文件格式。
- (15) ISO/IEC 14496-15AVC文件格式, 定义了基于第十二部分的用于存储第十部分的视频内容的文件格式。
- (16) ISO/IEC 14496-16动画框架扩展AFX(Animation Framework eXtension)。
- (17) ISO/IEC 14496-17同步文本字幕格式。
- (18) ISO/IEC 14496-18字体压缩和流式传输(针对公开字体格式)。
- (19) ISO/IEC 14496-19合成材质流(Synthesized Texture Stream)。
- (20) ISO/IEC 14496-20简单场景表示(LASer for Lightweight Scene Representation)。
- (21) ISO/IEC 14496-21用于描绘(Rendering)的MPEG-J拓展。
- (22) ISO/IEC 14496-22开放字体格式(Open Font Format)。
- (23) ISO/IEC 14496-2符号化音乐表示(Symbolic Music Representation)。
- (24) ISO/IEC 14496-24音频与系统交互作用(Audio and systems interaction)。
- (25) ISO/IEC 14496-253D图形压缩模型(3D Graphics Compression Model)。
- (26) ISO/IEC 14496-26音频一致性检查: 定义了测试音频数据与ISO/IEC 14496-3是否一致的方法(Audio conformance)。
- (27) ISO/IEC 14496-273D图形一致性检查: 定义了测试3D图形数据与ISO/IEC 14496-11:2005, ISO/IEC 14496-16:2006, ISO/IEC 14496-21:2006, 和 ISO/IEC 14496-25:2009是否一致的方法(3D Graphics conformance)。

参考

http://blog.csdn.net/yu_yuan_1314/article/details/9406587

<http://m.blog.chinaunix.net/uid-26009923-id-5702652.html>

视频格式说明

<http://www.zhihu.com/question/20997688>

atom说明

http://wiki.multimedia.cx/?title=QuickTime_container

<http://atomicparsley.sourceforge.net/mpeg-4files.html>

<http://www.ftyps.com/what.html>

http://blog.csdn.net/yu_yuan_1314/article/details/9366703

http://blog.csdn.net/yu_yuan_1314/article/details/9078287

开源库


qtfaststart python编写的, 如果moov在文件尾部, 移动到文件头部

<https://github.com/danielgtaylor/qtfaststart.git>

from : <https://www.jianshu.com/p/dd6b8aa625d4>

分类: 音视频文件



 DoubleLi
关注 - 29
粉丝 - 2080


+加关注

« 上一篇: HTTP协议中的chunked编码解析

» 下一篇: 关于ES、PES、PS/TS 码流

posted on 2020-08-03 17:04 DoubleLi 阅读(616) 评论(0) 编辑 收藏 举报

[刷新评论](#) [刷新页面](#) [返回顶部](#)

 登录后才能查看或发表评论, 立即 [登录](#) 或者 [逛逛](#) 博客园首页



App开发者高效成长

增长变现闭环

收入提升 **28%**

立即注册

编辑推荐：

- 聊聊我在微软外服的工作经历及一些个人见解
- 死磕 NIO — Reactor 模式就一定意味着高性能吗？
- 消息队列那么多，为什么建议深入了解下RabbitMQ？
- 技术管理进阶——管人还是管事？
- 以终为始：如何让你的开发符合预期

最新新闻：

- 何小鹏：争取2024年实现飞行汽车量产 价格100万以内 (2021-10-24 23:35)
 - 供应链危机提振美国在线二手市场 全年销售额预计超650亿美元 (2021-10-24 22:00)
 - CityTree：一款利用苔藓和机器学习来捕捉空气污染的设备 (2021-10-24 20:53)
 - 1024程序员节各家怎么过：送霸王洗发水、集体穿格子衫、盲人按摩 (2021-10-24 20:00)
 - 新卫星图展示泰国季风洪水所带来的巨大影响 (2021-10-24 19:00)
- » 更多新闻...

历史上的今天：

- 2017-08-03 用msys1.0+mingw gcc4.7.1编译libodb(C++ ORM)
- 2017-08-03 ODB学习笔记之基础环境搭建
- 2017-08-03 ORM:ODB安装使用过程
- 2017-08-03 HTTP方式播放FLV/mp4：nginx+Yamdi/MP4BOX
- 2017-08-03 搭建 Http Dynamic Streaming 点播/直播服务器
- 2017-08-03 RTMP HLS HTTP 直播协议一次看个够
- 2017-08-03 用VLC做流媒体服务器