郭风朴(guofengpu)的Android影音专栏

IT路上的牛耕实践者

转 hls之m3u8、ts流格式详解

2017年02月08日 09:20:08 阅读数：4569

HLS，Http Live Streaming 是由Apple公司定义的用于实时流传输的协议，HLS基于HTTP协议实现，传输内容包括两部分，一是M3U8描述文件，二是TS媒体文件。

1、M3U8文件

用文本方式对媒体文件进行描述，由一系列标签组成。

#EXTM3U

#EXT-X-TARGETDURATION:5

#EXTINF:5,

./0.ts

#EXTINF:5,

./1.ts

#EXTM3U：每个M3U8文件第一行必须是这个tag。

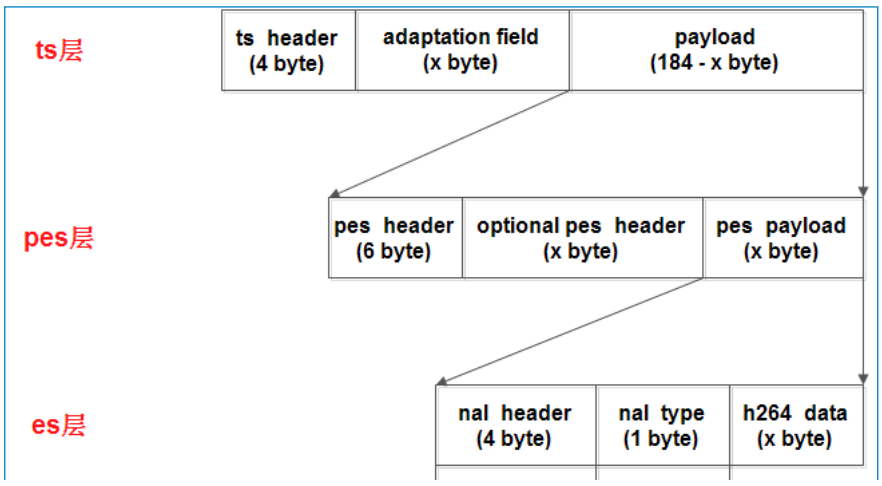
#EXT-X-TARGETDURATION：指定最大的媒体段时间长度（秒），#EXTINF中指定的时间长度必须小于或等于这个最大值。该值只能出现一次。

#EXTINF：描述单个媒体文件的长度。后面为媒体文件，如./0.ts

2、ts文件

ts文件为传输流文件，视频编码主要格式h264/mpeg4，音频为acc/MP3。

ts文件分为三层：ts层Transport Stream、pes层 Packet Elemental Stream、es层 Elementary Stream. es层就是音视频数据，pes层是在音视频数据上加了时间戳等对数据帧的说明信息，ts层就是在pes层加入数据流的识别和传输必须的信息



注： 详解如下

（1）ts层　ts包大小固定为188字节，ts层分为三个部分：ts header、adaptation field、payload。ts header固定4个字节；adaptation field可能存在也可能不存在，主要作用是给不足188字节的数据做填充；payload是pes数据。

ts header

sync_byte	8b	同步字节，固定为0x47
transport_error_indicator	1b	传输错误指示符，表明在ts头的adapt域后由一个无用字节，通常都为0，这个字节算在adapt域长度内
payload_unit_start_indicator	1b	负载单元起始标示符，一个完整的数据包开始时标记为1
transport_priority	1b	传输优先级，0为低优先级，1为高优先级，通常取0
pid	13b	pid值
transport_scrambling_control	2b	传输加扰控制，00表示未加密
adaptation_field_control	2b	是否包含自适应区，‘00’保留；‘01’为无自适应域，仅含有效负载；‘10’为仅含自适应域，无有效负载；‘11’为同时带有自适应域和有效负载。
continuity_counter	4b	递增计数器，从0-f，起始值不一定取0，但必须是连续的

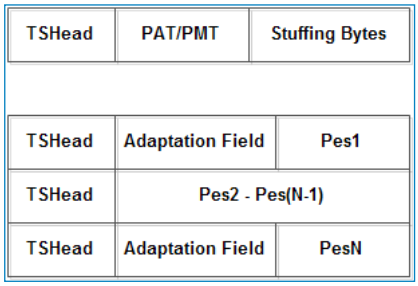
ts层的内容是通过PID值来标识的，主要内容包括：PAT表、PMT表、音频流、视频流。解析ts流要先找到PAT表，只要找到PAT就可以找到PMT，然后就可以找到音视频流了。PAT表的PID值固定为0。PAT表和PMT表需要定期插入ts流，因为用户随时可能加入ts流，这个间隔比较小，通常每隔几个视频帧就要加入PAT和PMT。PAT和PMT表是必须的，还可以加入其它表如SDT（业务描述表）等，不过hls流只要有PAT和PMT就可以播放了。

- PAT表：他主要的作用就是指明了PMT表的PID值。
- PMT表：他主要的作用就是指明了音视频流的PID值。
- 音频流/视频流：承载音视频内容。

adaption

adaptation_field_length	1B	自适应域长度，后面的字节数
flag	1B	取0x50表示包含PCR或0x40表示不包含PCR
PCR	5B	Program Clock Reference，节目时钟参考，用于恢复出与编码端一致的系统时序时钟STC（System Time Clock）。
stuffing_bytes	xB	填充字节，取值0xff

自适应区的长度要包含传输错误指示符标识的一个字节。pcr是节目时钟参考，pcr、dts、pts都是对同一个系统时钟的采样值，pcr是递增的，因此可以将其设置为dts值，音频数据不需要pcr。如果没有字段，ipad是可以播放的，但vlc无法播放。打包ts流时PAT和PMT表是没有adaptation field的，不够的长度直接补0xff即可。视频流和音频流都需要加adaptation field，通常加在一个帧的第一个ts包和最后一个ts包里，中间的ts包不加。



PAT格式

table_id	8b	PAT表固定为0x00
section_syntax_indicator	1b	固定为1

reserved	2b	固定为11
section_length	12b	后面数据的长度
transport_stream_id	16b	传输流ID，固定为0x0001
reserved	2b	固定为11
version_number	5b	版本号，固定为00000，如果PAT有变化则版本号加1
current_next_indicator	1b	固定为1，表示这个PAT表可以用，如果为0则要等待下一个PAT表
section_number	8b	固定为0x00
last_section_number	8b	固定为0x00
开始循环		
program_number	16b	节目号为0x0000时表示这是NIT，节目号为0x0001时,表示这是PMT
reserved	3b	固定为111
PID	13b	节目号对应内容的PID值
结束循环		
CRC32	32b	前面数据的CRC32校验码

PMT格式

table_id	8b	PMT表取值随意，0x02
section_syntax_indicator	1b	固定为1
zero	1b	固定为0
reserved	2b	固定为11
section_length	12b	后面数据的长度
program_number	16b	频道号码，表示当前的PMT关联到的频道，取值0x0001
reserved	2b	固定为11
version_number	5b	版本号，固定为00000，如果PAT有变化则版本号加1
current_next_indicator	1b	固定为1
section_number	8b	固定为0x00
last_section_number	8b	固定为0x00
reserved	3b	固定为111
PCR_PID	13b	PCR(节目参考时钟)所在TS分组的PID，指定为视频PID
reserved	4b	固定为1111
program_info_length	12b	节目描述信息，指定为0x000表示没有
开始循环		
stream_type	8b	流类型，标志是Video还是Audio还是其他数据，h.264编码对应0x1b，aac编码对应0x0f，mp3编码对应0x03
reserved	3b	固定为111
elementary_PID	13b	与stream_type对应的PID
reserved	4b	固定为1111
ES_info_length	12b	描述信息，指定为0x000表示没有
结束循环		
CRC32	32b	前面数据的CRC32校验码

（2）pes层

pes层是在每一个视频/音频帧上加入了时间戳等信息，pes包内容很多，我们只留下最常用的。

Pes Header (6B)	Optional Pes Header (3B -- 259B)	Payload (最大65526B)

pes start code	3B	开始码，固定为0x000001
stream id	1B	音频取值（0xc0-0xdf），通常为0xc0 视频取值（0xe0-0xef），通常为0xe0
pes packet length	2B	后面pes数据的长度，0表示长度不限制， 只有视频数据长度会超过0xffff
flag	1B	通常取值0x80，表示数据不加密、无优先级、备份的数据
flag	1B	取值0x80表示只含有pts，取值0xc0表示含有pts和dts
pes data length	1B	后面数据的长度，取值5或10
pts	5B	33bit值
dts	5B	33bit值

pts是显示时间戳、dts是解码时间戳，视频数据两种时间戳都需要，音频数据的pts和dts相同，所以只需要pts。有pts和dts两种时间戳是B帧引起的，I帧和P帧的pts等于dts。如果一个视频没有B帧，则pts永远和dts相同。从文件中顺序读取视频帧，取出的帧顺序和dts顺序相同。dts**算法**比较简单，初始值 + 增量即可，pts计算比较复杂，需要在dts的基础上加偏移量。

音频的pes中只有pts（同dts），视频的I、P帧两种时间戳都要有，视频B帧只要pts（同dts）。打包pts和dts就需要知道视频帧类型，但是通过容器格式我们是无法判断帧类型的，必须解析h.264内容才可以获取帧类型。

举例说明：

	I	P	B	B	B	P
读取顺序：	1	2	3	4	5	6
dts顺序：	1	2	3	4	5	6
pts顺序：	1	5	3	2	4	6

点播视频dts算法：

dts = 初始值 + 90000 / video_frame_rate，初始值可以随便指定，但是最好不要取0，video_frame_rate就是帧率，比如23、30。

pts和dts是以timescale为单位的，1s = 90000 time scale，一帧就应该是90000/video_frame_rate 个timescale。

用一帧的timescale除以采样频率就可以转换为一帧的播放时长

点播音频dts算法：

dts = 初始值 + (90000 * audio_samples_per_frame) / audio_sample_rate，audio_samples_per_frame这个值与编解码相关，aac取值1024，mp3取值1158，audio_sample_rate是采样率，比如24000、41000。AAC一帧解码出来是每声道1024个sample，也就是说一帧的时长为1024/sample_rate秒。所以每一帧时间戳依次0，1024/sample_rate，...，1024*n/sample_rate秒。

直播视频的dts和pts应该直接用直播数据流中的时间，不应该按公式计算。

（3）es层

es层指的就是音视频数据，我们只介绍h.264视频和aac音频。

h.264视频：

打包h.264数据我们必须给视频数据加上一个nalu（Network Abstraction Layer unit），nalu包括nalu header和nalu type，nalu header固定为0x00000001（帧开始）或0x000001（帧中）。h.264的数据是由slice组成的，slice的内容包括：视频、sps、pps等。nalu type决定了后面的h.264数据内容。

```
+---------------+
|0|1|2|3|4|5|6|7|
+-+-+-+-+-+-+-+-+
|F|NRI| Type |
+---------------+
```

F	1b	forbidden_zero_bit，h.264规定必须取0
NRI	2b	nal_ref_idc，取值0~3，指示这个nalu的重要性，I帧、sps、pps通常取3，P帧通常取2，B帧通常取0
Type	5b	参考下表

nal_unit_type	说明
0	未使用
1	非IDR图像片，IDR指关键帧
2	片分区A
3	片分区B
4	片分区C
5	IDR图像片，即关键帧
6	补充增强信息单元(SEI)
7	SPS序列参数集
8	PPS图像参数集
9	分解符
10	序列结束
11	码流结束
12	填充
13~23	保留
24~31	未使用

红色字体显示的内容是最常用的，打包es层数据时pes头和es数据之间要加入一个type=9的nalu，关键帧slice前必须要加入type=7和type=8的nalu，而且是紧邻。

转自：http://my.oschina.NET/u/727148/blog/666824

M3U8的简单介绍和在Android中使用的思路

(在项目中有用到m3u8，现在写篇博文，算是简单的总结

首先是名词介绍，什么是m3u8。m3u8是m3u的一种，不过是utf-8格式的，我记忆中说m3u8是苹果公司搞出来的一种播放的标准吧，其实简单来说就是把整个视频切成一段一段的，然后呢用一个m3u8格式来存这些个小段视频们的地址。可能大家就要问了，这么麻烦干嘛。其实m3u8是为了码率适配而生，而怎样去适配码率呢，这个下面介绍格式的时候会介绍到。

上两个m3u8文件的例子地址，大家能有直观的认识，这是我从Vitamio的官网上扒的。

我总结了一下我遇到的m3u8格式，虽然不能说涵盖了全部的情况，但是也差不多了：

1、一级目录（我觉着一级的目录没有适配码率的功能）

1.1、打开第一级m3u8文件，能找到真正的视频地址

1.2、第一级m3u8文件中，没有真正的视频地址，需要拼接才能找到真正的视频地址

2、二级目录

2.1、二级地址在一级文件中直接能看到

2.2、二级地址在一级文件中不能直接看到，需要拼接一级链接的地址才能找到二级文件的地址

2.2、打开二级目录，能找到整整的视频地址

2.3、没有真正的视频地址，需要拼接才能找到真正的视频地址

篇幅关系我不能给大家全部列举出这些全部的可能性。我就拿最麻烦的举个例子，其他的大家自行脑补吧，原理都是一样的，怎么样都跑不出协议的范畴之外。

我们在浏览器中输入**http://devimages.apple.com/iphone/samples/bipbop/bipbopall.m3u8**，会得到一个名为**bipbopall.m3u8**的文件，此文件的内容如下：

#EXTM3U

#EXT-X-STREAM-INF:PROGRAM-ID=1,BANDWIDTH=200000

gear1/prog_index.m3u8

#EXT-X-STREAM-INF:PROGRAM-ID=1,BANDWIDTH=311111

gear2/prog_index.m3u8

#EXT-X-STREAM-INF:PROGRAM-ID=1,BANDWIDTH=484444

gear3/prog_index.m3u8

#EXT-X-STREAM-INF:PROGRAM-ID=1,BANDWIDTH=737777

gear4/prog_index.m3u8

这就符合上面的2.2种情况，这四种码率的m3u8的地址你都不能直接得到，那怎么办呢，我们用得到这个文件的链接地址的前半段**http://devimages.apple.com/iphone/samples/bipbop/**拼接上二级文件的相对地址**gear1/prog_index.m3u8**得到一个地址**http://devimages.apple.com/iphone/samples/bipbop/gear1/prog_index.m3u8**。

把此地址放到浏览器中，我们又会得到一个同样名为prog_index.m3u8的文件，内容如下：

#EXTM3U

#EXT-X-TARGETDURATION:10

#EXT-X-MEDIA-SEQUENCE:0

#EXTINF:10, no desc

fileSequence0.ts

#EXTINF:10, no desc

fileSequence1.ts

#EXTINF:10, no desc

fileSequence2.ts

#EXTINF:10, no desc

#EXTINF:10, no desc

fileSequence4.ts

#EXTINF:10, no desc

fileSequence5.ts

#EXTINF:10, no desc

.

.

.

#EXTINF:10, no desc

fileSequence179.ts

#EXTINF:1, no desc

fileSequence180.ts

#EXT-X-ENDLIST

我们很开心的发现，这设计简直是...ng)夺(xin)天(bing)工(kuang)，我们还是没有得到真正的视频地址，老办法拼接后我们得到这么一段链**http://devimages.apple.com/i.../samples/bipbop/gear1/fileSequence179.ts**，这就是真正的视频地址。

我举的这个例子是最复杂的情况，...的情况对于这个来说都是相对简单的。就跟软件设计一样，我们先考虑到最难得情况，简单的来说就迎刃而解了

此篇博文没有具体介绍m3u8的格式，...位看管不了解的话还请自行Google之。

此篇博文完全是作者的经验之谈，...有不确切的地方还请见谅，转载请贴原文地址。

文章标签： hls 解码 H264

个人分类： RTMP/HTTP-FLV/RTSP/Internet