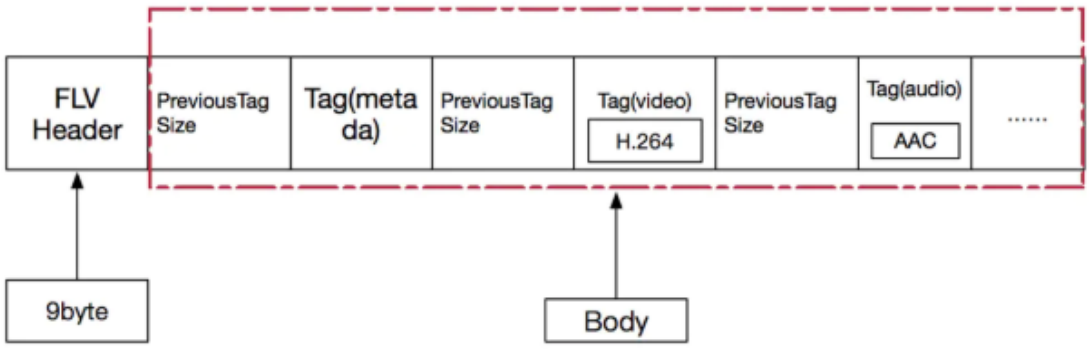
FLV封装原理

1.FLV介绍

FLV（Flash Video）是Adobe公司设计开发的一种流行的流媒体格式，由于其视频文件体积轻巧、封装简单等特点，使其很适合在互联网上进行应用。此外，FLV可以使用Flash Player进行播放，而Flash Player插件已经安装在全世界绝大部分浏览器上，这使得通过网页播放FLV视频十分容易。目前主流的视频网站如优酷网，土豆网，乐视网等网站无一例外地使用了FLV格式。FLV封装格式的文件后缀通常为“.flv”。

FLV封装格式是由一个文件头(flie header)和 文件体(file Body)组成。其中，FLV body由一对对的(Previous Tag Size字段 + tag)组成。Previous Tag Size字段 排列在Tag之前，占用4个字节。Previous Tag Size记录了前面一个Tag的大小，用于逆向读取处理。FLV header后的第一个Pervious Tag Size的值为0。Tag一般可以分为3种类型：脚本(帧)数据类型、音频数据类型、视频数据。FLV数据以大端序进行存储，在解析时需要注意。一个标准FLV文件结构如下图：



FLV文件结构

https://blog.csdn.net/guoyunfei123

FLV文件的详细内容结构如下图

<table>
<tr><td rowspan="4">Flv Header</td><td colspan="3">Signature（3 字节）为文件标识，总为"FLV"，（0x46，0x4c，0x66）</td></tr>
<tr><td colspan="3">Version（1 字节）为版本，目前为 0x01</td></tr>
<tr><td colspan="3">Flags（1 字节）前 5 位保留，必须为 0。第 6 位表示是否存在音频 Tag。第 7 位保留，必须为 0。第 8 位表示是否存在视频 Tag。</td></tr>
<tr><td colspan="3">Headersize（4 字节）为从 File Header 开始到 File Body 开始的字节数，版本 1 中总为 9。</td></tr>
<tr><td rowspan="10">Flv Body</td><td colspan="3">Previous Tag Size #0（4 字节）表示前一个 Tag 的长度</td></tr>
<tr><td rowspan="6">Tag #1</td><td rowspan="4">Tag Header</td><td>Type（1 字节）表示 Tag 类型，包括音频（0x08），视频（0x09）和 script data（0x12），其他类型值被保留</td></tr>
<tr><td>Datasize（3 字节）表示该 Tag Ddata 部分的大小</td></tr>
<tr><td>Timestamp（3 字节）表示该 Tag 的时间戳</td></tr>
<tr><td>Timestamp_ex（1 字节）表示时间戳的扩展字节，当 24 位数值不够时，该字节最为最高位将时间戳扩展为 32 位数值</td></tr>
<tr><td>StreamID（3 字节）表示 stream id 总是 0</td></tr>
<tr><td>Tag Data</td><td>不同类型 Tag 的 data 部分结构各不相同，当 header 的结构是相同的</td></tr>
<tr><td colspan="3">Previous Tag size #1 即 Tag #1 的大小（11 + Datasize）</td></tr>
<tr><td colspan="3">Tag #2</td></tr>
<tr><td colspan="3">Previous Tag size #2</td></tr>
<tr><td colspan="3">... ...</td></tr>
<tr><td colspan="3">Tag #N</td></tr>
<tr><td colspan="3">Previous Tag size #N</td></tr>
</table>

https://blog.csdn.net/guoyunfei123

2.FLV Header

下面的数据type中，UI表示无符号整形，后面跟的数字表示其长度是多少位。比如UI8，表示无法整形，长度一个字节。UI24是三个字节,UI[8*n]表示多个字节。UB表示位域，UB5表示一个字节的5位。可以参考c中的位域结构体。

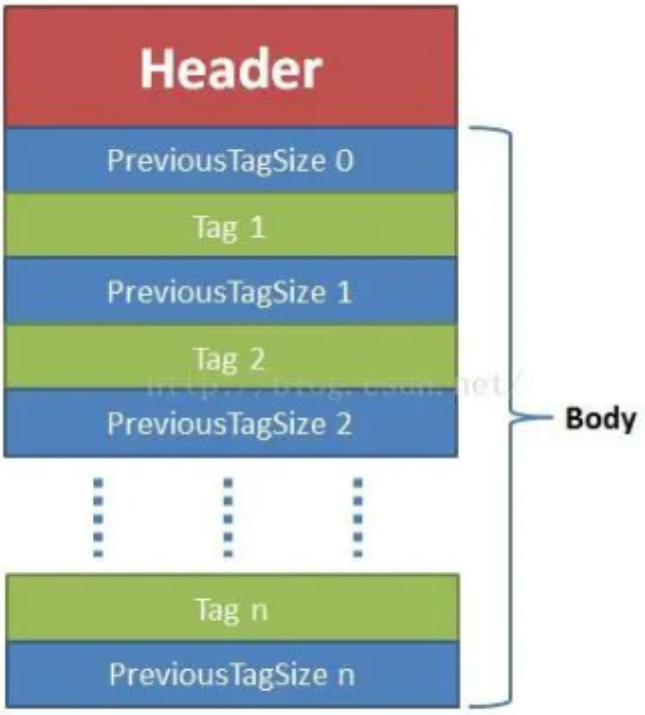
FLV头占9个字节，用来标识文件为FLV类型，以及后续存储的音视频流。一个FLV文件，每种类型的tag都属于一个流，也就是一个flv文件最多只有一个音频流，一个视频流，不存在多个独立的音视频流在一个文件的情况。FLV头的结构如下：

字段	类型	说明
Signature	UI8	'F' (0x46)
Signature	UI8	'L' (0x4C)
Signature	UI8	'V' (0x56)
Version	UI8	FLV 版本。例如，0x01 表示 FLV 版本 1
TypeFlags	UI8	b[0] 是否存在视频流 b[2] 是否存在音频流 其他字段保留，值为0
DataOffset	UI32	FLV Header 长度(字节)

https://blog.csdn.net/guoyunfei123

3.FLV Body

FLV Header之后，就是FLV File Body.FLV File Body是由一连串的back-pointers + tags构成。Back-pointer表示Previous Tag Size(前一个tag的字节数据长度)，占4个字节。



FLV Body结构

https://blog.csdn.net/guoyunfei123

4.FLV Tag

每一个Tag也是由两部分组成:tag header 和 tag data。Tag Header里存放的是当前tag的类型、数据区(tag data)的长度等信息。tag header一般占11个字节的内存空间。FLV tag结构如下：

Field	Type	Comment
Tag类型	UI8	8:audeo 9:video 18:Script data(脚本数据) all Others:reserved 其他所有值未使用
数据区大小	UI24	当前tag的数据区的大小，不包含包头
时戳	UI24	当前帧时戳，单位是毫秒。相对值，第一个tag的时戳总是为0
时戳扩展字段	UI8	如果时戳大于0xFFFFFF，将会使用这个字节。这个字节是时戳的高8位，上面的三个字节是低24位。
StreamID	UI24	总是为0
数据区	UI[8*n]	数据区数据

https://blog.csdn.net/guoyunfei123

FLV Tag的类型可以是视频、音频和Script(脚本类型)，下面分别介绍这三种Tag类型

4.1 Script Tag Data结构(脚本类型、帧类型)

该类型Tag又被称为MetaData Tag,存放一些关于FLV视频和音频的元信息，比如：duration、width、height等。通常该类型Tag会作为FLV文件的第一个tag，并且只有一个，跟在File Header后。该类型Tag DaTa的结构如下所示:

帧类型 Tag Data结构

第一个AMF包：
第1个字节表示AMF包类型，一般总是0x02，表示字符串。第2-3个字节为UI16类型值，标识字符串的长度，一般总是0x000A（“onMetaData”长度）。后面字节为具体的字符串，一般总为“onMetaData”（6F,6E,4D,65,74,61,44,61,74,61）。

第二个AMF包：
第1个字节表示AMF包类型，一般总是0x08，表示数组。第2-5个字节为UI32类型值，表示数组元素的个数。后面即为各数组元素的封装，数组元素为元素名称和值组成的对。常见的数组元素如下表所示。

值	Comment
duration	时长
width	视频宽度
heiht	视频高度
video data rate	视频码率
frame rate	视频帧率
video codec id	视频编码方式
audio sample rate	音频采样率
audio sample size	音频采样精度
stereo	是否为立体声
audio codec id	音频编码方式
filesize	文件大小
...	...

https://blog.csdn.net/guoyunfei123

4.2 Audio Tag Data结构(音频类型)

音频Tag Data区域开始的第一个字节包含了音频数据的参数信息，从第二个字节开始为音频流数据。结构如下：

第一个字节为音频的信息，格式如下：

Field	Type	Comment
音频格式	UB4	0 = Linear PCM, platform endian 1 =ADPCM 2 = MP3 3 = Linear PCM, little endian 4 = Nellymoser 16-kHz mono 5 = Nellymoser 8-kHz mono 6 = Nellymoser 7 = G.711 A-law logarithmic PCM 8 = G.711 mu-law logarithmic PCM 9 = reserved 10 = AAC 11 = Speex 14 = MP3 8-Khz 15 = Device-specific sound flv是不支持g711a的，如果要用，可能要用线性音频。
采样率	UB2	0 = 5.5-kHz 1 = 11-kHz 2 = 22-kHz 3 = 44-kHz 对于AAC总是3。由此可以看出FLV封装格式并不支持48KHz的采样率
采样精度	UB1	0 = snd8Bit 1 = snd16Bit 压缩过的音频都是16bit
音频声道	UB1	0 = sndMono 单声道 1 = sndStereo 立体声，双声道 对于AAC总是1

https://blog.csdn.net/guoyunfei123

格式 3，linear PCM，存储原始 PCM 采样点。如果采样位深为 8，采样点数据为无符号型。如果采样位深为 16，采样点数据为小端存储的带符号型。如果是立体声，左右声道采样点交织存放：左-右-左-右-...

格式 0 与格式 3 的不同之处只有一点：格式 0 存储 16 位采样数据，采用的大小端顺序是创建 FLV 文件的平台所使用的大小端顺序。因此，不应使用格式 0，而应使用格式 3。

格式 4 (Nellymoser 16-kHz mono) 和格式 5 (Nellymoser 8 kHz mono)，是两种特殊情况, 因为采样率字段无法表示 8 kHz 和 16 kHz。当采样格式是格式 4 或格式 5 时，Flash 播放器会忽略采样率和声音类型两个字段。对于其他采样率的 Nellymoser 格式, 即格式 6，则正常使用采样率和声音类型两个字段。

格式 10，AAC，声音类型应为 1 (立体声) 且采样率应为 3 (44 kHz)。这并不表示 FLV 中的 AAC 音频总是立体声、44 kHz的数据。实际上，Flash 播放器会忽略这两个值，而从已编码的 AAC 位流中提取出声道数和采样率信息。

格式 11，Speex，音频以 16 kHz采样率压缩为单声道，采样率字段值应为 0，采样位深字段值应为 1，声音类型字段值应为 0。

格式 7，8，14 和 15 保留。

第二个字节开始为音频数据

Field	Type	Comment
音频数据	UI[8*n]	如果是PCM线性数据，存储的时候每个16bit小端存储，有符号。 如果音频格式是AAC，则存储的数据是AAC AUDIO DATA，否则为线性数组。

4.3 Video Tag Data结构(视频类型)

视频Tag Data开始的第一个字节包含视频数据的参数信息，从第二个字节开始为视频流数据。结构如下：

video Tag Data结构:

Field	Type	Comment
帧类型	UB4	1: keyframe (for AVC, a seekable frame)——h264的IDR，关键帧，可重入帧。 2: inter frame (for AVC, a non- seekable frame)——h264的普通帧 3: disposable inter frame (H.263 only) 4: generated keyframe (reserved for server use only) 5: video info/command frame
编码ID	UB4	使用哪种编码类型： 1: JPEG (currently unused) 2: Sorenson H.263 3: Screen video 4: On2 VP6 5: On2 VP6 with alpha channel 6: Screen video version 2 7: AVC

https://blog.csdn.net/guoyunfei123

第二个字节开始为视频数据

Field	Type	Comment
视频数据	UI[8*n]	如果是avc，则参考下面的介绍：AVC VIDEO PACKET