**FMP4Player设计文档**

**类逻辑结构图**

s

类FMP4Player

Media Source

类RemuxController

track

data

H.264

类BufferController

类AACParser

类H264Parser

类H264Remuxer

类AACRemuxer

AAC

类BufferController

track

data

WebSocket

流媒体服务器

HTML5

<video/>

**类说明：**

* FMP4Player:MSE播放器类，在HTML5页面，用户直接调用FMP4Player的函数。该类封装了所有的播放功能，是与用户直接交互也是唯一需要交互的类；
* RemuxController:H.264码流和AAC码流封装类，把从服务器接收的H.264数据和AAC数据封装成fMP4格式的数据，添加到<video/>进行播放；该类内部包含了封装H.264数据的类H264Remuxer对象和封装AAC数据的类AACRemuxer；
* H264Remuxer:封装H.264数据的类，把纯H.264数据封装为fMP4需要的格式；该类内部包含了H.264数据解析的类H264Parser;
* H264Parser:解析H.264的NALU数据；
* AACRemuxer：封装AAC数据的类，把AAC数据封装为fMP4需要的格式；该类内部包含了AAC数据解析的类AACParser;
* AACParser:解析AAC的ADTS数据；

**数据包格式**

服务器向FMP4Player传输的数据包，必须符合一定的格式，FMP4Player才可进行解封装、重封装进行播放，数据包的格式如下：

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| AAC Duration (ms) | H.264 Duration(ms) | AAC Data length | AAC Data | H.264 Data |

AAC Duration:AAC数据需要播放多长时间，4字节长度，小端字节序，单位为毫秒；

H.264 Duration:H.264数据需要播放多长时间，4字节长度，小端字节序，单位为毫秒；

AAC Data length:AAC数据的长度，4字节长度，小端字节序，单位为字节；

AAC Data:AAC 数据，二进制数据；该数据必须为完整的AAC帧数据，否则播放器会产生不可预知问题；

H.264 Data:H.264数据，二进制数据；该数据必须为完整的H.264帧数据，否则播放器会产生不可预知问题；

格式对播放器有一定要求：

1. 若mode设置为video，则不需要遵从该格式，直接将h264数据传递给播放器，带格式会导致视频无法播放；
2. 若mode设置为audio，则不需要遵从该格式，直接将aac数据传递给播放器，带格式会导致音频无法播放；
3. 若mode设置为both，表明该数据即含有视频数据，又含有音频数据，需要按照指定格式填充数据；

**SourceBuffer初始化**

在MediaSource对象与<video/>标签关联之后，播放器接收服务器发送的数据包，解析并提取其中的H.264数据和AAC数据，然后使用解析得到的SPS和ADTS进行SourceBuffer的初始化。初始化结束后，将初始化得到的SourceBuffer添加到MediaSource中，开始进行视频和音频的播放。

**播放器初始化参数**

播放器初始化参数为JSON对象，其描述如下：

|  |  |  |
| --- | --- | --- |
| **参数** | **类型** | **说明** |
| node | 字符串或者<video/>对象 | 播放关联的<video/>对象ID或者<video/>对象 |
| mode | 字符串 | 播放模式，为以下三个值之一：   1. ‘video’; 视频 2. ‘audio’; 音频 3. ‘both’; 两者   当传入’both’时，feed函数需要同时传入视频和音频，否则播放失败 |
| debug | 布尔型 | 播放器是否以debug模式运行；当播放器以debug模式运行时，会打印调试信息；否则不打印  true 或者 false |
| flushingBufferTime | 整数 | 播放缓存的时间，单位毫秒。  播放器播放时，需要把音视频数据存储到播放缓存里，当超过flushingBufferTime时，播放缓存的音视频并清空缓存，如果flushingBufferTime为0，则立即播放 |
| videoTagMaxDelay | 整数 | 最大延迟时间，单位毫秒。  当<video/>播放时，内部会缓存数据，如果缓存延迟超过videoTagMaxDelay毫秒，定时器会调整<video/>当前播放进度,跳过延迟的数据 |
| autoClearSourceBuffer | 布尔型 | 是否自动清空SourceBuffer的缓存 |
| fps | 整数 | FPS，可以设置为0。系统feed函数的duration计算每一帧的播放时间。如果duration参数未设置，则系统根据FPS进行计算每一帧的播放时间，如果位置为0，表示使用默认值，默认值为30。 |
| readFpsFromTrack | 布尔型 | 是否从track里解析fps值。该参数只对视频’video’使用。当设置为true时，系统解析H.264的SPS，获取fps值。 |
| onReady | 函数 | 回调函数。MSE初始化成功完成之后，调用该回调 |
| onError | 函数 | 回调函数。当SourceBuffer遇到错误时，调用该回调函数 |

**清空<video/>缓存**

在构造函数中，有清空<video/>缓存的延迟时间参数videoTagMaxDelay（单位毫秒）。当<video/> MediaSource的音频SourceBuffer或者视频SourceBuffer初始化成功之后，每隔flushingBufferTime毫秒（如果flushingBufferTime为0，则默认为1000），系统检查<video/>是否有缓存。如果有缓存并且缓存时间超过videoTagMaxDelay毫秒，则调整<video/>的播放时间为当前时间，以此来达到清空<video/>缓存的目标。

MediaSource的SourceBuffer初始化成功的条件：

1. 如果构造函数参数mode为‘video’，则当H.264 SPS和PPS解析成功后，并且系统支持该H.264数据格式，则视频SourceBuffer初始化成功，即MediaSource的SourceBuffer初始化成功；
2. 如果构造函数参数mode为‘audio’，则当ADTS解析成功后，并且系统支持该AAC数据格式，则音频SourceBuffer初始化成功，即MediaSource的SourceBuffer初始化成功；
3. 如果构造函数参数mode为‘both’，则当H.264 SPS和PPS以及AAC ADTS解析成功后，并且系统支持该H.264数据格式和AAC数据格式，则视频SourceBuffer和音频SourceBuffer初始化成功，即MediaSource的SourceBuffer初始化成功。**注意**：必须视频SourceBuffer和音频SourceBuffer都初始化成功之后，MediaSource的SourceBuffer才算初始化成功。两者缺一不可。

**flushingBufferTime说明**

构造函数参数flushingBufferTime(单位毫秒)为系统缓存flush时间。播放器接收到音视频数据会存储到播放器缓存内。如果flushingBufferTime为0，播放器将缓存内的音视频数据通过SourceBuffer的appendBuffer函数立即播放；如果flushingBufferTime大于0，例如500，则播放器会每隔500毫秒将缓存内的音视频数据通过SourceBuffer的appendBuffer函数进行播放。由此也就产生了500毫秒的视频时延。

**视频播放流程**

见文档《视频播放流程.vsdx》