

原 XBMC源代码分析 7：视频播放器（dvdplayer）-输入流（以libRTMP为例）

2014年01月10日 01:05:43 阅读数：8166

前文分析了XBMC的基本结构：

[XBMC源代码分析 1：整体结构以及编译方法](#)

[XBMC源代码分析 2：Addons（皮肤Skin）](#)

[XBMC源代码分析 3：核心部分（core）-综述](#)

[XBMC源代码分析 4：视频播放器（dvdplayer）-解码器（以ffmpeg为例）](#)

[XBMC源代码分析 5：视频播放器（dvdplayer）-解复用器（以ffmpeg为例）](#)

[XBMC源代码分析 6：视频播放器（dvdplayer）-文件头（以ffmpeg为例）](#)

本文我们分析XBMC中视频播放器（dvdplayer）中的输入流部分。由于输入流种类很多，因此以RTMP输入流为例进行分析。

XBMC中输入流部分文件目录结构如下图所示。

从目录中文件的名称我们可以看出，XBMC支持多种输入方式：File，HTSP，HTTP，RTMP等等。在这里我们看看RTMP部分的源代码。对应DVDInputStreamRTMP.h和DVDInputStreamRTMP.cpp

先来看看DVDInputStreamRTMP.h

```
[cpp]
1.  /*
2.   * 雷霄骅
3.   *  leixiaohua1020@126.com
4.   *  中国传媒大学/数字电视技术
5.   *
6.   */
7.  //如果有libRTMP
8.  #ifndef HAS_LIBRTMP
9.
10. #include "DVDInputStream.h"
11. #include "DllLibRTMP.h"
12. //支持RTMP输入流的类，继承CDVDInputStream
13. class CDVDInputStreamRTMP
14. : public CDVDInputStream
15. , public CDVDInputStream::ISearchTime
16. {
17. public:
18.     CDVDInputStreamRTMP();
19.     virtual ~CDVDInputStreamRTMP();
20.     virtual bool    Open(const char* strFile, const std::string &content); //打开
21.     virtual void    Close(); //关闭
22.     virtual int     Read(uint8_t* buf, int buf_size); //读取
23.     virtual int64_t Seek(int64_t offset, int whence); //跳转到
24.     bool            SeekTime(int iTimeInMsec);
25.     virtual bool    Pause(double dTime); //暂停
26.     virtual bool    IsEOF();
27.     virtual int64_t GetLength();
28.
29.     CCriticalSection m_RTMPSection;
30.
31. protected:
32.     bool    m_eof;
33.     bool    m_bPaused;
34.     char*    m_sStreamPlaying;
35.     std::vector<CStdString> m_optionvalues;
36.
37.     RTMP    *m_rtmp;
38.     DllLibRTMP m_libRTMP;
39. };
40.
41. #endif
```

该类中包含了Open(), Close(), Read(), Seek(), Pause() 这类的方法。实现了对RTMP协议的各种操作。这些方法都是CDVDInputStreamRTMP父类CDVDInputStream中的方法。可以看一下CDVDInputStream的定义，就知道了。

```
[cpp]
1.  //输入流类
2.  class CDVDInputStream
```

```

3. {
4. public:
5.     class IChannel
6.     {
7.     public:
8.         virtual ~IChannel() {};
9.         virtual bool NextChannel(bool preview = false) = 0;
10.        virtual bool PrevChannel(bool preview = false) = 0;
11.        virtual bool SelectChannelByNumber(unsigned int channel) = 0;
12.        virtual bool SelectChannel(const PVR::CPVRChannel &channel) { return false; };
13.        virtual bool GetSelectedChannel(PVR::CPVRChannelPtr&) { return false; };
14.        virtual bool UpdateItem(CFileItem& item) = 0;
15.        virtual bool CanRecord() = 0;
16.        virtual bool IsRecording() = 0;
17.        virtual bool Record(bool bOnOff) = 0;
18.        virtual bool CanPause() = 0;
19.        virtual bool CanSeek() = 0;
20.    };
21.
22.    class IDisplayTime
23.    {
24.    public:
25.        virtual ~IDisplayTime() {};
26.        virtual int GetTotalTime() = 0;
27.        virtual int GetTime() = 0;
28.    };
29.
30.    class ISeekTime
31.    {
32.    public:
33.        virtual ~ISeekTime() {};
34.        virtual bool SeekTime(int ms) = 0;
35.    };
36.
37.    class IChapter
38.    {
39.    public:
40.        virtual ~IChapter() {};
41.        virtual int GetChapter() = 0;
42.        virtual int GetChapterCount() = 0;
43.        virtual void GetChapterName(std::string& name) = 0;
44.        virtual bool SeekChapter(int ch) = 0;
45.    };
46.
47.    class IMenus
48.    {
49.    public:
50.        virtual ~IMenus() {};
51.        virtual void ActivateButton() = 0;
52.        virtual void SelectButton(int iButton) = 0;
53.        virtual int GetCurrentButton() = 0;
54.        virtual int GetTotalButtons() = 0;
55.        virtual void OnUp() = 0;
56.        virtual void OnDown() = 0;
57.        virtual void OnLeft() = 0;
58.        virtual void OnRight() = 0;
59.        virtual void OnMenu() = 0;
60.        virtual void OnBack() = 0;
61.        virtual void OnNext() = 0;
62.        virtual void OnPrevious() = 0;
63.        virtual bool OnMouseMove(const CPoint &point) = 0;
64.        virtual bool OnMouseClicked(const CPoint &point) = 0;
65.        virtual bool IsInMenu() = 0;
66.        virtual void SkipStill() = 0;
67.        virtual double GetTimeStampCorrection() = 0;
68.        virtual bool GetState(std::string &xmlstate) = 0;
69.        virtual bool SetState(const std::string &xmlstate) = 0;
70.
71.    };
72.
73.    class ISeekable
74.    {
75.    public:
76.        virtual ~ISeekable() {};
77.        virtual bool CanSeek() = 0;
78.        virtual bool CanPause() = 0;
79.    };
80.
81.    enum ENextStream
82.    {
83.        NEXTSTREAM_NONE,
84.        NEXTSTREAM_OPEN,
85.        NEXTSTREAM_RETRY,
86.    };
87.
88.    CDVDInputStream(DVDStreamType m_streamType);
89.    virtual ~CDVDInputStream();
90.    virtual bool Open(const char* strFileName, const std::string& content); //打开
91.    virtual void Close() = 0; //关闭
92.    virtual int Read(uint8_t* buf, int buf_size) = 0; //读取
93.    virtual int64_t Seek(int64_t offset, int whence) = 0; //跳转

```

```

94.     virtual bool Pause(double dTime) = 0; //暂停
95.     virtual int64_t GetLength() = 0;
96.     virtual std::string& GetContent() { return m_content; };
97.     virtual std::string& GetFileName() { return m_strFileName; }
98.     virtual CURL &GetURL() { return m_url; }
99.     virtual ENextStream NextStream() { return NEXTSTREAM_NONE; }
100.    virtual void Abort() {}
101.    virtual int GetBlockSize() { return 0; }
102.    virtual void ResetScanTimeout(unsigned int iTimeoutMs) {}
103.
104.    /*! \brief Indicate expected read rate in bytes per second.
105.     * This could be used to throttle caching rate. Should
106.     * be seen as only a hint
107.     */
108.    virtual void SetReadRate(unsigned rate) {}
109.
110.    /*! \brief Get the cache status
111.     \return true when cache status was successfully obtained
112.     */
113.    virtual bool GetCacheStatus(XFILE::SCacheStatus *status) { return false; }
114.
115.    bool IsStreamType(DVDStreamType type) const { return m_streamType == type; }
116.    virtual bool IsEOF() = 0;
117.    virtual BitstreamStats GetBitstreamStats() const { return m_stats; }
118.
119.    void SetFileItem(const CFileItem& item);
120.
121.protected:
122.    DVDStreamType m_streamType;
123.    std::string m_strFileName;
124.    CURL m_url;
125.    BitstreamStats m_stats;
126.    std::string m_content;
127.    CFileItem m_item;
128.};

```

回到CDVDInputStreamRTMP类本身。可以看一下Open(), Close(), Read(), Seek(), Pause()这些方法的函数体。这些方方通过调用libRTMP中相应的方法，完成了对RTMP流媒体的各种操作。

```

[cpp]
1.  /*
2.   * 雷霄骅
3.   * leixiaohua1020@126.com
4.   * 中国传媒大学/数字电视技术
5.   *
6.   */
7.  //打开
8.  bool CDVDInputStreamRTMP::Open(const char* strFile, const std::string& content)
9.  {
10.     if (m_sStreamPlaying)
11.     {
12.         free(m_sStreamPlaying);
13.         m_sStreamPlaying = NULL;
14.     }
15.
16.     if (!CDVDInputStream::Open(strFile, "video/x-flv"))
17.         return false;
18.
19.     CSingleLock lock(m_RTMPSection);
20.
21.     // libRTMP can and will alter strFile, so take a copy of it
22.     m_sStreamPlaying = (char*)calloc(strlen(strFile)+1, sizeof(char));
23.     strcpy(m_sStreamPlaying, strFile);
24.     //libRTMP中的设置URL
25.     if (!m_libRTMP.SetupURL(m_rtmp, m_sStreamPlaying))
26.         return false;
27.
28.     // SetOpt and SetAVal copy pointers to the value. librtmp doesn't use the values until the Connect() call,
29.     // so value objects must stay allocated until then. To be extra safe, keep the values around until Close(),
30.     // in case librtmp needs them again.
31.     m_optionvalues.clear();
32.     for (int i=0; options[i].name; i++)
33.     {
34.         CStdString tmp = m_item.GetProperty(options[i].name).asString();
35.         if (!tmp.empty())
36.         {
37.             m_optionvalues.push_back(tmp);
38.             AVal av_tmp;
39.             SetAVal(av_tmp, m_optionvalues.back());
40.             m_libRTMP.SetOpt(m_rtmp, &options[i].key, &av_tmp);
41.         }
42.     }
43.     //建立RTMP链接中的NetConnection和NetStream
44.     if (!m_libRTMP.Connect(m_rtmp, NULL) || !m_libRTMP.ConnectStream(m_rtmp, 0))
45.         return false;
46.

```

```

47.     m_eof = false;
48.
49.     return true;
50. }
51. //关闭
52. // close file and reset everything
53. void CDVDInputStreamRTMP::Close()
54. {
55.     CSingleLock lock(m_RTMPSection);
56.     CDVDInputStream::Close();
57.     //关闭连接
58.     m_libRTMP.Close(m_rtmp);
59.
60.     m_optionvalues.clear();
61.     m_eof = true;
62.     m_bPaused = false;
63. }
64. //读取
65. int CDVDInputStreamRTMP::Read(uint8_t* buf, int buf_size)
66. { //读取
67.     int i = m_libRTMP.Read(m_rtmp, (char *)buf, buf_size);
68.     if (i < 0)
69.         m_eof = true;
70.
71.     return i;
72. }
73. //跳转到
74. int64_t CDVDInputStreamRTMP::Seek(int64_t offset, int whence)
75. {
76.     if (whence == SEEK_POSSIBLE)
77.         return 0;
78.     else
79.         return -1;
80. }
81. //暂停
82. bool CDVDInputStreamRTMP::Pause(double dTime)
83. {
84.     CSingleLock lock(m_RTMPSection);
85.
86.     m_bPaused = !m_bPaused;
87.
88.     CLog::Log(LOGNOTICE, "RTMP Pause %s requested", m_bPaused ? "TRUE" : "FALSE");
89.
90.     m_libRTMP.Pause(m_rtmp, m_bPaused);
91.
92.     return true;
93. }

```

版权声明：本文为博主原创文章，未经博主允许不得转载。<https://blog.csdn.net/leixiaohua1020/article/details/17512667>

文章标签：[xbmc](#) [librtmp](#) [源代码](#) [播放器](#) [输入](#)

个人分类：[XBMC](#) [libRTMP](#)

所属专栏：[开源多媒体项目源代码分析](#)

此PDF由spygg生成,请尊重原作者版权!!!

我的邮箱:liushidc@163.com