

原 Media Player Classic - HC 源代码分析 3：核心类（CMainFrame）（2）

2013年10月28日 23:52:56 阅读数：5853

Media Player Classic - HC 源代码分析系列文章列表：

[Media Player Classic - HC 源代码分析 1：整体结构](#)

[Media Player Classic - HC 源代码分析 2：核心类（CMainFrame）（1）](#)

[Media Player Classic - HC 源代码分析 3：核心类（CMainFrame）（2）](#)

[Media Player Classic - HC 源代码分析 4：核心类（CMainFrame）（3）](#)

[Media Player Classic - HC 源代码分析 5：关于对话框（CAboutDlg）](#)

[Media Player Classic - HC 源代码分析 6：MediaInfo选项卡（CPPageFileMediaInfo）](#)

[Media Player Classic - HC 源代码分析 7：详细信息选项卡（CPPageFileInfoDetails）](#)



上一篇文章分析了Media Player Classic - HC (mpc-hc)的源代码中的核心类 CMainFrame： [Media Player Classic - HC 源代码分析 2:核心类（CMainFrame）（1）](#)

主要介绍了CMainFrame类中的以下几个函数（“->”代表调用关系）：

OpenMedia() -> OpenMediaPrivate() -> OpenFile()

本文补充介绍CMainFrame类中的其他一些函数。

再回顾一下打开文件功能主要所在的函数OpenMediaPrivate()：

```
[cpp]
1. //打开一个媒体 (private)
2. bool CMainFrame::OpenMediaPrivate(CAutoPtr<OpenMediaData> pOMD)
3. {
4.     //获得设置信息
5.     CAppSettings& s = AfxGetAppSettings();
6.
7.     if (m_iMediaLoadState != MLS_CLOSED) {
8.         ASSERT(0);
9.         return false;
10.    }
11.    //OpenFileData
12.    //OpenDVDDData
13.    //OpenDeviceData
14.    //里面包含了文件或者DVD信息（名称等）
15.    OpenFileData* pFileData = dynamic_cast<OpenFileData*>(pOMD.m_p);
16.    OpenDVDDData* pDVDDData = dynamic_cast<OpenDVDDData*>(pOMD.m_p);
17.    OpenDeviceData* pDeviceData = dynamic_cast<OpenDeviceData*>(pOMD.m_p);
18.    if (!pFileData && !pDVDDData && !pDeviceData) {
19.        ASSERT(0);
20.        return false;
21.    }
22.
23.    // Clear DXVA state ...
24.    ClearDXVAState();
25.
26.    #ifdef _DEBUG
27.        // Debug trace code - Begin
28.        // Check for bad / buggy auto loading file code
29.        if (pFileData) {
30.            POSITION pos = pFileData->fns.GetHeadPosition();
31.            UINT index = 0;
32.            while (pos != nullptr) {
33.                CString path = pFileData->fns.GetNext(pos);
34.                TRACE( T("CMainFrame::OpenMediaPrivate: pFileData->fns.GetNext(%d) = %s", index, path) );
```

```

34.         TRACE(_T(TEXT("mainFrame::OpenMediaPrivate - pFileData->fns[%d]"), index);
35.         TRACE(_T(TEXT("\t%s\n"), path.GetString())); // %ws - wide character string always
36.         index++;
37.     }
38. }
39. // Debug trace code - End
40. #endif
41.
42. CString mi_fn = _T("");
43.
44. if (pFileData) {
45.     if (pFileData->fns.IsEmpty()) {
46.         return false;
47.     }
48.
49.     CString fn = pFileData->fns.GetHead();
50.
51.     int i = fn.Find(_T(":\\"));
52.     if (i > 0) {
53.         CString drive = fn.Left(i + 2);
54.         UINT type = GetDriveType(drive);
55.         CAtlList<CString> sl;
56.         if (type == DRIVE_REMOVABLE || type == DRIVE_CDROM && GetCDROMType(drive[0], sl) != CDROM_Audio) {
57.             int ret = IDRETRY;
58.             while (ret == IDRETRY) {
59.                 WIN32_FIND_DATA findFileData;
60.                 HANDLE h = FindFirstFile(fn, &findFileData);
61.                 if (h != INVALID_HANDLE_VALUE) {
62.                     FindClose(h);
63.                     ret = IDOK;
64.                 } else {
65.                     CString msg;
66.                     msg.Format(IDS_MAINFRM_114, fn);
67.                     ret = AfxMessageBox(msg, MB_RETRYCANCEL);
68.                 }
69.             }
70.
71.             if (ret != IDOK) {
72.                 return false;
73.             }
74.         }
75.         mi_fn = fn;
76.     }
77. }
78.
79. SetLoadState(MLS_LOADING);
80.
81. // FIXME: Don't show "Closed" initially
82. PostMessage(WM_KICKIDLE);
83.
84. CString err;
85.
86. m_fUpdateInfoBar = false;
87. BeginWaitCursor();
88.
89. try {
90.     CComPtr<IVMRMixerBitmap9> pVMB;
91.     CComPtr<IMFVideoMixerBitmap> pMFVMB;
92.     CComPtr<IMadVRTextOsd> pMVTO;
93.     if (m_fOpeningAborted) {
94.         throw (UINT)IDS_AG_ABORTED;
95.     }
96.
97.     OpenCreateGraphObject(pOMD);
98.
99.     if (m_fOpeningAborted) {
100.         throw (UINT)IDS_AG_ABORTED;
101.     }
102.
103.     SetupIViAudReg();
104.
105.     if (m_fOpeningAborted) {
106.         throw (UINT)IDS_AG_ABORTED;
107.     }
108.     //按类型的不同打开不同的文件
109.     if (pFileData) {
110.         //文件
111.         OpenFile(pFileData);
112.     } else if (pDVDDData) {
113.         //DVD
114.         OpenDVD(pDVDDData);
115.     } else if (pDeviceData) {
116.         if (s.iDefaultCaptureDevice == 1) {
117.             HRESULT hr = OpenBDAGraph();
118.             if (FAILED(hr)) {
119.                 throw (UINT)IDS_CAPTURE_ERROR_DEVICE;
120.             }
121.         } else {
122.             OpenCapture(pDeviceData);
123.         }
124.     } else {
125.         throw (UINT)IDS_INVALID_PARAMS_ERROR;

```

```

126.     }
127.
128.     m_pCAP2 = nullptr;
129.     m_pCAP = nullptr;
130.     //查找接口
131.     m_pGB->FindInterface(__uuidof(ISubPicAllocatorPresenter), (void**)&m_pCAP, TRUE);
132.     m_pGB->FindInterface(__uuidof(ISubPicAllocatorPresenter2), (void**)&m_pCAP2, TRUE);
133.     m_pGB-
>FindInterface(__uuidof(IVMRWindowlessControl9), (void**)&m_pVMRWC, FALSE); // might have IVMRMixerBitmap9, but not IVMRWindowlessCon
rol9
134.     m_pGB->FindInterface(__uuidof(IVMRMixerControl9), (void**)&m_pVMRMC, TRUE);
135.     m_pGB->FindInterface(__uuidof(IVMRMixerBitmap9), (void**)&pVMB, TRUE);
136.     m_pGB->FindInterface(__uuidof(IMFVideoMixerBitmap), (void**)&pMFVMB, TRUE);
137.     pMVT0 = m_pCAP;
138.
139.     if (s.fShowOSD || s.fShowDebugInfo) { // Force OSD on when the debug switch is used
140.         if (pVMB) {
141.             m OSD.Start(m_pVideoWnd, pVMB, IsD3DFullScreenMode());
142.         } else if (pMFVMB) {
143.             m OSD.Start(m_pVideoWnd, pMFVMB, IsD3DFullScreenMode());
144.         } else if (pMVT0) {
145.             m OSD.Start(m_pVideoWnd, pMVT0);
146.         }
147.     }
148.     //VMR9
149.     SetupVMR9ColorControl();
150.
151.     // === EVR !
152.     m_pGB->FindInterface(__uuidof(IMFVideoDisplayControl), (void**)&m_pMFVDC, TRUE);
153.     m_pGB->FindInterface(__uuidof(IMFVideoProcessor), (void**)&m_pMFVP, TRUE);
154.     if (m_pMFVDC) {
155.         m_pMFVDC->SetVideoWindow(m_pVideoWnd->m_hWnd);
156.     }
157.
158.     //SetupEVRColorControl();
159.     //does not work at this location
160.     //need to choose the correct mode (IMFVideoProcessor::SetVideoProcessorMode)
161.
162.     BeginEnumFilters(m_pGB, pEF, pBF) {
163.         if (m_pLN21 == pBF) {
164.             m_pLN21->SetServiceState(s.fClosedCaptions ? AM_L21_CCSTATE_On : AM_L21_CCSTATE_Off);
165.             break;
166.         }
167.     }
168.     EndEnumFilters;
169.
170.     if (m_fOpeningAborted) {
171.         throw (UINT)IDS_AG_ABORTED;
172.     }
173.     //打开自定义的Graph
174.     OpenCustomizeGraph();
175.
176.     if (m_fOpeningAborted) {
177.         throw (UINT)IDS_AG_ABORTED;
178.     }
179.     //设置视频窗口
180.     OpenSetupVideo();
181.
182.     if (m_fOpeningAborted) {
183.         throw (UINT)IDS_AG_ABORTED;
184.     }
185.     //设置音量
186.     OpenSetupAudio();
187.
188.     if (m_fOpeningAborted) {
189.         throw (UINT)IDS_AG_ABORTED;
190.     }
191.
192.     if (m_pCAP && (!m_fAudioOnly || m_fRealMediaGraph)) {
193.
194.         if (s.fDisableInternalSubtitles) {
195.             m_pSubStreams.RemoveAll(); // Needs to be replaced with code that checks for forced subtitles.
196.         }
197.
198.         m_posFirstExtSub = nullptr;
199.         POSITION pos = pOMD->subs.GetHeadPosition();
200.         while (pos) {
201.             LoadSubtitle(pOMD->subs.GetNext(pos), nullptr, true);
202.         }
203.     }
204.
205.     if (m_fOpeningAborted) {
206.         throw (UINT)IDS_AG_ABORTED;
207.     }
208.     //设置视频窗口标题
209.     OpenSetupWindowTitle(pOMD->title);
210.
211.     if (s.fEnableEDLEditor) {
212.         m_wndEditListEditor.OpenFile(pOMD->title);
213.     }
214.

```

```

215.         if (::GetCurrentThreadId() == AfxGetApp()->m_nThreadID) {
216.             OnFilePostOpenMedia();
217.         } else {
218.             PostMessage(WM_COMMAND, ID_FILE_POST_OPENMEDIA);
219.         }
220.
221.         while (m_iMediaLoadState != MLS_LOADED
222.             && m_iMediaLoadState != MLS_CLOSING // FIXME
223.             ) {
224.             Sleep(50);
225.         }
226.         //设置音频流
227.         DWORD audstm = SetupAudioStreams();
228.         //设置字幕流
229.         DWORD substm = SetupSubtitleStreams();
230.
231.         if (audstm) {
232.             OnPlayAudio(ID_AUDIO_SUBITEM_START + audstm);
233.         }
234.         if (substm) {
235.             SetSubtitle(substm - 1);
236.         }
237.
238.         // PostMessage instead of SendMessage because the user might call CloseMedia and then we would deadlock
239.
240.         PostMessage(WM_COMMAND, ID_PLAY_PAUSE);
241.
242.         m_bFirstPlay = true;
243.
244.         if (!(s.nCLSwitches & CLSW_OPEN) && (s.nLoops > 0)) {
245.             PostMessage(WM_COMMAND, ID_PLAY_PLAY);
246.         } else {
247.             // If we don't start playing immediately, we need to initialize
248.             // the seekbar and the time counter.
249.             OnTimer(TIMER_STREAMPOSPOLLER);
250.             OnTimer(TIMER_STREAMPOSPOLLER2);
251.         }
252.
253.         s.nCLSwitches &= ~CLSW_OPEN;
254.
255.         if (pFileData) {
256.             if (pFileData->rtStart > 0) {
257.                 PostMessage(WM_RESUMEFROMSTATE, (WPARAM)PM_FILE, (LPARAM)(pFileData-
258. >rtStart / 10000)); // REFERENCE_TIME doesn't fit in LPARAM under a 32bit env.
259.             }
260.             } else if (pDVDDData) {
261.                 if (pDVDDData->pDvdState) {
262.                     PostMessage(WM_RESUMEFROMSTATE, (WPARAM)PM_DVD, (LPARAM)(CComPtr<IDvdState>(pDVDDData->pDvdState).Detach())); // m
ust be released by the called message handler
263.                 }
264.             } else if (pDeviceData) {
265.                 m_wndCaptureBar.m_capdlg.SetVideoInput(pDeviceData->vinut);
266.                 m_wndCaptureBar.m_capdlg.SetVideoChannel(pDeviceData->vchannel);
267.                 m_wndCaptureBar.m_capdlg.SetAudioInput(pDeviceData->ainput);
268.             }
269.         } catch (LPCTSTR msg) {
270.             err = msg;
271.         } catch (CString& msg) {
272.             err = msg;
273.         } catch (UINT msg) {
274.             err.LoadString(msg);
275.         }
276.
277.         EndWaitCursor();
278.
279.         if (!err.IsEmpty()) {
280.             //关闭
281.             CloseMediaPrivate();
282.             m_closingmsg = err;
283.
284.             if (err != ResStr(IDS_AG_ABORTED)) {
285.                 if (pFileData) {
286.                     m_wndPlaylistBar.SetCurValid(false);
287.
288.                     if (m_wndPlaylistBar.IsAtEnd()) {
289.                         m_nLoops++;
290.                     }
291.
292.                     if (s.fLoopForever || m_nLoops < s.nLoops) {
293.                         bool hasValidFile = false;
294.
295.                         if (m_nLastSkipDirection == ID_NAVIGATE_SKIPBACK) {
296.                             hasValidFile = m_wndPlaylistBar.SetPrev();
297.                         } else {
298.                             hasValidFile = m_wndPlaylistBar.SetNext();
299.                         }
300.
301.                         if (hasValidFile) {
302.                             OpenCurPlaylistItem();
303.                         }
304.                     } else if (m_wndPlaylistBar.GetCount() > 1) {

```

```

304.         DoAfterPlaybackEvent();
305.     }
306. } else {
307.     OnNavigateSkip(ID_NAVIGATE_SKIPFORWARD);
308. }
309. }
310. } else {
311.     m_wndPlaylistBar.SetCurValid(true);
312.
313.     // Apply command line audio shift
314.     if (s.rtShift != 0) {
315.         SetAudioDelay(s.rtShift);
316.         s.rtShift = 0;
317.     }
318. }
319.
320. m_nLastSkipDirection = 0;
321.
322. if (s.AutoChangeFullscrRes.bEnabled && (m_fFullScreen || IsD3DFullScreenMode())) {
323.     AutoChangeMonitorMode();
324. }
325. if (m_fFullScreen && s.fRememberZoomLevel) {
326.     m_fFirstFSAfterLaunchOnFS = true;
327. }
328.
329. m_LastOpenFile = pOMD->title;
330.
331. PostMessage(WM_KICKIDLE); // calls main thread to update things
332.
333. if (!m_bIsBDPlay) {
334.     m_MPLSPlaylist.RemoveAll();
335.     m_LastOpenBDPath = _T("");
336. }
337. m_bIsBDPlay = false;
338.
339. return err.IsEmpty();
340. }

```

来看一看OpenMediaPrivate()函数的细节：

1.开始的时候有这么一句

```

1. CAppSettings& s = AfxGetAppSettings();

```

在这里涉及到一个类CAppSettings，存储的是mpc-hc用到的各种设置信息。源代码如下：

```

1. //应用程序中的各种参数
2. class CAppSettings
3. {
4.     bool fInitialized;
5.
6.     class CRecentFileAndURLList : public CRecentFileList
7.     {
8.     public:
9.         CRecentFileAndURLList(UINT nStart, LPCTSTR lpszSection,
10.                                LPCTSTR lpszEntryFormat, int nSize,
11.                                int nMaxDisplen = AFX_ABBREV_FILENAME_LEN);
12.
13.         virtual void Add(LPCTSTR lpszPathName); // we have to override CRecentFileList::Add because the original version can't handle
14.         URLs
15.     };
16. public:
17.     bool fShaderEditorWasOpened;
18.
19.     // cmdline params
20.     UINT nCLSwitches;
21.     CATLList<CString> slFiles, slDubs, slSubs, slFilters;
22.
23.     // Initial position (used by command line flags)
24.     REFERENCE_TIME rtShift;
25.     REFERENCE_TIME rtStart;
26.     ULONG lDVDTitle;
27.     ULONG lDVDChapter;
28.     DVD_HMSF_TIMECODE DVDPosition;
29.
30.     CSize sizeFixedWindow;
31.     bool HasFixedWindowSize() const { return sizeFixedWindow.cx > 0 || sizeFixedWindow.cy > 0; }
32.     //int iFixedWidth, iFixedHeight;
33.     int iMonitor;
34.
35.     CString ParseFileName(CString const& param);
36.     void ParseCommandLine(CATLList<CString>& cmdln);
37.

```

```

37.
38. // Added a Debug display to the screen (/debug option)
39. bool          fShowDebugInfo;
40. int           iAdminOption;
41.
42.
43. //播放器 Player
44. bool          fAllowMultipleInst;
45. bool          fTrayIcon;
46. bool          fShowOSD;
47. bool          fLimitWindowProportions;
48. bool          fSnapToDesktopEdges;
49. bool          fHideCDROMsSubMenu;
50. DWORD         dwPriority;
51. int           iTitleBarTextStyle;
52. bool          fTitleBarTextTitle;
53. bool          fKeepHistory;
54. CRecentFileAndURLList MRU;
55. CRecentFileAndURLList MRUDub;
56. CFilePositionList filePositions;
57. CDVDPositionList dvdPositions;
58. bool          fRememberDVDPos;
59. bool          fRememberFilePos;
60. bool          bRememberPlaylistItems;
61. bool          fRememberWindowPos;
62. CRect         rcLastWindowPos;
63. bool          fRememberWindowSize;
64. bool          fSavePnSZoom;
65. double        dZoomX;
66. double        dZoomY;
67.
68. // Formats
69. CMediaFormats m_Formats;
70. bool          fAssociatedWithIcons;
71.
72. // Keys
73. CList<wmcmd> wmcmds;
74. HACCEL       hAccel;
75. bool          fWinLirc;
76. CString      strWinLircAddr;
77. CWinLircClient WinLircClient;
78. bool          fUIIce;
79. CString      strUIIceAddr;
80. CUIceClient  UIIceClient;
81. bool          fGlobalMedia;
82.
83. //图标 Logo
84. UINT         nLogoId;
85. bool          fLogoExternal;
86. CString      strLogoFileName;
87.
88. //web界面? Web Interface
89. BOOL          fEnableWebServer;
90. int           nWebServerPort;
91. int           nCmdLnWebServerPort;
92. bool          fWebServerUseCompression;
93. bool          fWebServerLocalhostOnly;
94. bool          fWebServerPrintDebugInfo;
95. CString      strWebRoot, strWebDefIndex;
96. CString      strWebServerCGI;
97.
98. //播放时候 Playback
99. int           nVolume;
100. bool          fMute;
101. int           nBalance;
102. int           nLoops;
103. bool          fLoopForever;
104. bool          fRewind;
105. bool          fRememberZoomLevel;
106. int           nAutoFitFactor;
107. int           iZoomLevel;
108. CStringW      strAudiosLanguageOrder;
109. CStringW      strSubtitlesLanguageOrder;
110. bool          fEnableWorkerThreadForOpening;
111. bool          fReportFailedPins;
112. bool          fAutoloadAudio;
113. bool          fAutoloadSubtitles;
114. bool          fBlockVSTFilter;
115. UINT          nVolumeStep;
116. UINT          nSpeedStep;
117.
118. // DVD/OGM
119. bool          fUseDVDPath;
120. CString       strDVDPath;
121. LCID          idMenuLang, idAudioLang, idSubtitlesLang;
122. bool          fAutoSpeakerConf;
123. bool          fClosedCaptions;
124.
125. //输出 Output
126. CRenderersSettings m_RenderersSettings;
127. int           iDSVideoRendererType;
128. int           iRMVideoRendererTvne;

```

```

129.         int                iQTVideoRendererType;
130.
131.         CStringW            strAudioRendererDisplayName;
132.         bool                fD3DFullscreen;
133.
134.         //全屏 Fullscreen
135.         bool                fLaunchfullscreen;
136.         bool                fShowBarsWhenFullScreen;
137.         int                 nShowBarsWhenFullScreenTimeOut;
138.         bool                fExitFullScreenAtTheEnd;
139.         CStringW            strFullScreenMonitor;
140.         AChFR               AutoChangeFullscrRes;
141.         bool                fRestoreResAfterExit;
142.
143.         // Sync Renderer Settings
144.
145.         // Capture (BDA configuration)
146.         int                 iDefaultCaptureDevice;        // Default capture device (analog=0, 1=digital)
147.         CString             strAnalogVideo;
148.         CString             strAnalogAudio;
149.         int                 iAnalogCountry;
150.         CString             strBDANetworkProvider;
151.         CString             strBDATuner;
152.         CString             strBDAReceiver;
153.         //CString            strBDAStandard;
154.         int                 iBDAScanFreqStart;
155.         int                 iBDAScanFreqEnd;
156.         int                 iBDABandwidth;
157.         bool                fBDAUseOffset;
158.         int                 iBDAOffset;
159.         bool                fBDAIgnoreEncryptedChannels;
160.         UINT                nDVBLastChannel;
161.         CATllist<CDVBChannel> m_DVBChannels;
162.         DVB_RebuildFilterGraph nDVBRebuildFilterGraph;
163.         DVB_StopFilterGraph nDVBSStopFilterGraph;
164.
165.         // Internal Filters
166.         bool                SrcFilters[SRC_LAST + !SRC_LAST];
167.         bool                TraFilters[TRA_LAST + !TRA_LAST];
168.
169.         //音频 Audio Switcher
170.         bool                fEnableAudioSwitcher;
171.         bool                fAudioNormalize;
172.         UINT                nAudioMaxNormFactor;
173.         bool                fAudioNormalizeRecover;
174.         UINT                nAudioBoost;
175.         bool                fDownSampleTo441;
176.         bool                fAudioTimeShift;
177.         int                 iAudioTimeShift;
178.         bool                fCustomChannelMapping;
179.         int                 nSpeakerChannels;
180.         DWORD               pSpeakerToChannelMap[AS_MAX_CHANNELS][AS_MAX_CHANNELS];
181.
182.         // External Filters
183.         CAutoPtrList<FilterOverride> m_filters;
184.
185.         //字幕 Subtitles
186.         bool                fOverridePlacement;
187.         int                 nHorPos, nVerPos;
188.         int                 nSubDelayInterval;
189.
190.         // Default Style
191.         STSStyle            subdefstyle;
192.
193.         // Misc
194.         bool                bPreferDefaultForcedSubtitles;
195.         bool                fPrioritizeExternalSubtitles;
196.         bool                fDisableInternalSubtitles;
197.         bool                bAllowOverridingExternalSplitterChoice;
198.         CString             strSubtitlePaths;
199.         CString             strISDb;
200.
201.         // Tweaks
202.         int                 nJumpDistS;
203.         int                 nJumpDistM;
204.         int                 nJumpDistL;
205.         bool                fFastSeek;
206.         bool                fShowChapters;
207.         bool                bNotifySkype;
208.         bool                fPreventMinimize;
209.         bool                fUseWin7TaskBar;
210.         bool                fLCDSupport;
211.         bool                fUseSearchInFolder;
212.         bool                fUseTimeTooltip;
213.         int                 nTimeTooltipPosition;
214.         CString             strOSDFont;
215.         int                 nOSDSize;
216.
217.         //亮度色度饱和度 Miscellaneous
218.         int                 iBrightness;
219.         int                 iContrast;

```

```

220.     int             iHue;
221.     int             iSaturation;
222.     int             nUpdaterAutoCheck;
223.     int             nUpdaterDelay;
224.
225.     // MENUS
226.     // View
227.     int             iCaptionMenuMode; // normal -> hidemenu -> frameonly -> borderless
228.     bool            fHideNavigation;
229.     UINT            nCS; // Control state for toolbars
230.     // Language
231.     LANGID          language;
232.     // Subtitles menu
233.     bool            fEnableSubtitles;
234.     bool            fUseDefaultSubtitlesStyle;
235.     // Video Frame
236.     int             iDefaultVideoSize;
237.     bool            fKeepAspectRatio;
238.     CSize           sizeAspectRatio;
239.     bool            fCompMonDeskARDiff;
240.     // Pan&Scan
241.     CString         strPnSPreset;
242.     CStringArray    m_pnspresets;
243.     // On top menu
244.     int             iOnTop;
245.     // After Playback
246.     bool            fExitAfterPlayback;
247.     bool            fNextInDirAfterPlayback;
248.
249.     // WINDOWS
250.     // Add Favorite
251.     bool            bFavRememberPos;
252.     bool            bFavRelativeDrive;
253.     // Save Image...
254.     CString         strSnapshotPath, strSnapshotExt;
255.     // Save Thumbnails...
256.     int             iThumbRows, iThumbCols, iThumbWidth;
257.     // Shader Editor
258.     struct Shader {
259.         CString     label;
260.         CString     target;
261.         CString     srcdata;
262.     };
263.     CAtlList<Shader> m_shaders;
264.     // Shader Combiner
265.     bool            fToggleShader;
266.     bool            fToggleShaderScreenSpace;
267.     CString         strShaderList;
268.     CString         strShaderListScreenSpace;
269.     // Playlist (contex menu)
270.     bool            bShufflePlaylistItems;
271.     bool            bHidePlaylistFullScreen;
272.
273.     // OTHER STATES
274.     CStringW        strLastOpenDir;
275.     UINT            nLastWindowType;
276.     UINT            nLastUsedPage;
277.     bool            fRemainingTime;
278.     bool            fLastFullScreen;
279.
280.     bool            fIntRealMedia;
281.     //bool            fRealMediaRenderless;
282.     //float           dRealMediaQuickTimeFPS;
283.     //int             iVideoRendererType;
284.     //int             iQuickTimeRenderer;
285.     //bool            fMonitorAutoRefreshRate;
286.     bool            fEnableEDLEditor;
287.
288.     HWND            hMasterWnd;
289.
290.     bool            IsD3DFullscreen() const;
291.     CString         SelectedAudioRenderer() const;
292.     bool            IsISREnabled() const;
293.
294. private:
295.     CString         SrcFiltersKeys[SRC_LAST + !SRC_LAST];
296.     CString         TraFiltersKeys[TRA_LAST + !TRA_LAST];
297.
298.     __int64         ConvertTimeToMSec(const CString& time) const;
299.     void            ExtractDVDStartPos(CString& strParam);
300.
301.     void            CreateCommands();
302.
303.     void            SaveExternalFilters(CAutoPtrList<FilterOverride>& filters, LPCTSTR baseKey = IDS_R_EXTERNAL_FILTERS);
304.     void            LoadExternalFilters(CAutoPtrList<FilterOverride>& filters, LPCTSTR baseKey = IDS_R_EXTERNAL_FILTERS);
305.     void            ConvertOldExternalFiltersList();
306.
307.     void            UpdateRenderersData(bool fSave);
308.     friend void     CRenderersSettings::UpdateData(bool bSave);
309.
310. public:

```



```

311.     CAppSettings();
312.     virtual ~CAppSettings();
313.
314.     void         SaveSettings();
315.     void         LoadSettings();
316.     void         SaveExternalFilters() { if (fInitialized) { SaveExternalFilters(m_filters); } };
317.
318.     void         GetFav(favtype ft, CAtlList<CString>& sl) const;
319.     void         SetFav(favtype ft, CAtlList<CString>& sl);
320.     void         AddFav(favtype ft, CString s);
321.
322.     CDVBChannel* FindChannelByPref(int nPrefNumber);
323.
324.     bool         GetAllowMultiInst() const;
325.
326.     static bool   IsVSFilterInstalled();
327.     static bool   HasEVR();
328. };

```

由代码可见，包含的参数信息很多。在mpc-hc中，任何需要获取设置信息的地方，都可以使用AfxGetAppSettings()获得CAppSettings的引用。

2.OpenSetupVideo()这个函数的作用是设置视频窗口，源代码如下：

```

1. //设置视频窗口
2. void CMainFrame::OpenSetupVideo()
3. {
4.     //大部分都在确定：m_fAudioOnly是否为True
5.     m_fAudioOnly = true;
6.     //获得视频的宽和高，然后调整窗口大小
7.     if (m_pMFVDC) { // EVR
8.         m_fAudioOnly = false;
9.     } else if (m_pCAP) {
10.         CSize vs = m_pCAP->GetVideoSize();
11.         m_fAudioOnly = (vs.cx <= 0 || vs.cy <= 0);
12.     } else {
13.         {
14.             long w = 0, h = 0;
15.
16.             if (CComQIPtr<IBasicVideo> pBV = m_pGB) {
17.                 pBV->GetVideoSize(&w, &h);
18.             }
19.
20.             if (w > 0 && h > 0) {
21.                 m_fAudioOnly = false;
22.             }
23.         }
24.         //如果 m_fAudioOnly=true;再检查
25.         if (m_fAudioOnly) {
26.             BeginEnumFilters(m_pGB, pEF, pBF) {
27.                 long w = 0, h = 0;
28.
29.                 if (CComQIPtr<IVideoWindow> pVW = pBF) {
30.                     long lVisible;
31.                     if (FAILED(pVW->get_Visible(&lVisible))) {
32.                         continue;
33.                     }
34.
35.                     pVW->get_Width(&w);
36.                     pVW->get_Height(&h);
37.                 }
38.
39.                 if (w > 0 && h > 0) {
40.                     m_fAudioOnly = false;
41.                     break;
42.                 }
43.             }
44.             EndEnumFilters;
45.         }
46.     }
47.
48.     if (m_fShockwaveGraph) {
49.         m_fAudioOnly = false;
50.     }
51.
52.     if (m_pCAP) {
53.         SetShaders();
54.     }
55.     // else
56.     {
57.         // TESTME
58.         //设置所有者。。。
59.         m_pVW->put_Owner((OAHWND)m_pVideoWnd->m_hWnd);
60.         m_pVW->put_WindowStyle(WS_CHILD | WS_CLIPSIBLINGS | WS_CLIPCHILDREN);
61.         m_pVW->put_MessageDrain((OAHWND)m_hWnd);
62.
63.         for (CWnd* pWnd = m_wndView.GetWindow(GW_CHILD); pWnd; pWnd = pWnd->GetNextWindow()) {
64.             pWnd->EnableWindow(FALSE); // little trick to let WM_SETCURSOR thru
65.         }
66.     }
67.     //如果只有音频，则消灭视频窗口！
68.     if (m_fAudioOnly && IsD3DFullScreenMode()) {
69.         m_pFullscreenWnd->DestroyWindow();
70.     }
71. }

```

3. OpenSetupAudio()这个函数的作用是设置音频，源代码如下：

```

1. //设置音量
2. void CMainFrame::OpenSetupAudio()
3. {
4.     //设置音量
5.     m_pBA->put_Volume(m_wndToolBar.Volume);
6.
7.     // FIXME
8.     int balance = AfxGetAppSettings().nBalance;
9.
10.    int sign = balance > 0 ? -1 : 1; // -1: invert sign for more right channel
11.    if (balance > -100 && balance < 100) {
12.        balance = sign * (int)(100 * 20 * log10(1 - abs(balance) / 100.0f));
13.    } else {
14.        balance = sign * (-10000); // -10000: only left, 10000: only right
15.    }
16.    //设置均衡
17.    m_pBA->put_Balance(balance);
18. }

```

4.如果出现问题，则会调用CloseMediaPrivate()，关闭打开的媒体。

[cpp]  

```
1. //关闭
2. void CMainFrame::CloseMediaPrivate()
3. {
4.     SetLoadState(MLS_CLOSING); // why it before OnPlayStop()? // TODO: remake or add detailed comments
5.     OnPlayStop(); // SendMessage(WM_COMMAND, ID_PLAY_STOP);
6.     if (m_pMC) {
7.         m_pMC-
8. >Stop(); // needed for StreamBufferSource, because m_iMediaLoadState is always MLS_CLOSED // TODO: fix the opening for such media
9.     }
10.    SetPlaybackMode(PM_NONE);
11.    m_fLiveWM = false;
12.    m_fEndOfStream = false;
13.    m_rtDurationOverride = -1;
14.    m_kfs.RemoveAll();
15.    m_pCB.Release();
16.    {
17.        CAutoLock cAutoLock(&m_csSubLock);
18.        m_pSubStreams.RemoveAll();
19.    }
20.    m_pSubClock.Release();
21.
22.    //if (m_pVW) m_pVW->put_Visible(OAFALSE);
23.    //if (m_pVW) m_pVW->put_MessageDrain((OAHWND)NULL), m_pVW->put_Owner((OAHWND)NULL);
24.
25.    // IMPORTANT: IVMRSurfaceAllocatorNotify/IVMRSurfaceAllocatorNotify9 has to be released before the VMR/VMR9, otherwise it will cr
26.    in Release()
27.    //各种清空
28.    m_OSD.Stop();
29.    m_pCAP2.Release();
30.    m_pCAP.Release();
31.    m_pVMRWC.Release();
32.    m_pVMRMC.Release();
33.    m_pMFVP.Release();
34.    m_pMFVDC.Release();
35.    m_pLN21.Release();
36.    m_pSyncClock.Release();
37.
38.    m_pAMXBar.Release();
39.    m_pAMDF.Release();
40.    m_pAMVCCap.Release();
41.    m_pAMVCPprev.Release();
42.    m_pAMVSCCap.Release();
43.    m_pAMVSCPrev.Release();
44.    m_pAMASC.Release();
45.    m_pVidCap.Release();
46.    m_pAudCap.Release();
47.    m_pAMTuner.Release();
48.    m_pCGB.Release();
49.
50.    m_pDVDC.Release();
51.    m_pDVDI.Release();
52.    m_pAMOP.Release();
53.    m_pBI.Release();
54.    m_pQP.Release();
55.    m_pFS.Release();
56.    m_pMS.Release();
57.    m_pBA.Release();
58.    m_pBV.Release();
59.    m_pVW.Release();
60.    m_pME.Release();
61.    m_pMC.Release();
62.
63.    if (m_pGB) {
64.        m_pGB->RemoveFromROT();
65.        m_pGB.Release();
66.    }
67.
68.    m_pProv.Release();
69.
70.    m_fRealMediaGraph = m_fShockwaveGraph = m_fQuicktimeGraph = false;
71.
72.    m_VidDispName.Empty();
73.    m_AudDispName.Empty();
74.
75.    m_closingmsg.LoadString(IDS_CONTROLS_CLOSED);
76.
77.    AfxGetAppSettings().nCLSwitches &= CLSW_OPEN | CLSW_PLAY | CLSW_AFTERPLAYBACK_MASK | CLSW_NOFOCUS;
78.    //设置状态
79.    SetLoadState(MLS_CLOSED);
80. }
```

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/13297291>

文章标签：[mpc-hc](#) [源代码](#) [directshow](#) [播放器](#) [开源](#)

个人分类：[MPC-HC](#)

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