

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

2013年10月29日 00:34:53 阅读数：5855

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）](#)



此前已经写了2篇文章介绍Media Player Classic - HC（mpc-hc）源代码中的核心类 CMainFrame：

[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）](#)

此前的文章一直都是围绕着 OpenMedia()以及其调用的函数进行分析的。研究的都是和文件打开有关的功能。在这里再介绍一些其它函数。

在mpc-hc开始运行的时候，会调用OnCreate()：

```
[cpp]
1. <span style="font-family:Arial;font-size:12px;">刚打开的时候加载
2. int CMainFrame::OnCreate(LPCREATESTRUCT lpCreateStruct)
3. {
4.     if ( _super::OnCreate(lpCreateStruct) == -1) {
5.         return -1;
6.     }
7.     //加载菜单
8.     m_popup.LoadMenu(IDR_POPUP);
9.     m_popupmain.LoadMenu(IDR_POPUPMAIN);
10.
11.     // create a view to occupy the client area of the frame
12.     // 创建视频画面部分？
13.     if (!m_wndView.Create(nullptr, nullptr, AFX_WS_DEFAULT_VIEW,
14.         CRect(0, 0, 0, 0), this, AFX_IDW_PANE_FIRST, nullptr)) {
15.         TRACE(_T("Failed to create view window\n"));
16.         return -1;
17.     }
18.     // Should never be RTLed
19.     m_wndView.ModifyStyleEx(WS_EX_LAYOUTRTL, WS_EX_NOINHERITLAYOUT);
20.
21.     // static bars
22.     //各种状态栏
23.     BOOL bResult = m_wndStatusBar.Create(this);
24.     if (bResult) {
25.         bResult = m_wndStatsBar.Create(this);
26.     }
27.     if (bResult) {
28.         bResult = m_wndInfoBar.Create(this);
29.     }
30.     if (bResult) {
31.         bResult = m_wndToolBar.Create(this);
32.     }
33.     if (bResult) {
34.         bResult = m_wndSeekBar.Create(this);
35.     }
```

```

36.     if (!bResult) {
37.         TRACE(_T("Failed to create all control bars\n"));
38.         return -1;    // fail to create
39.     }
40.     // 各种Bar
41.     m_pFullscreenWnd = DEBUG_NEW CFullscreenWnd(this);
42.
43.     mBars.AddTail(&m_wndSeekBar);
44.     mBars.AddTail(&m_wndToolBar);
45.     mBars.AddTail(&m_wndInfoBar);
46.     mBars.AddTail(&m_wndStatsBar);
47.     mBars.AddTail(&m_wndStatusBar);
48.
49.     m_wndSeekBar.Enable(false);
50.
51.     // dockable bars
52.     // 可停靠的
53.     EnableDocking(CBRS_ALIGN_ANY);
54.
55.     m_dockingbars.RemoveAll();
56.
57.     m_wndSubresyncBar.Create(this, AFX_IDW_DOCKBAR_TOP, &m_csSubLock);
58.     m_wndSubresyncBar.SetBarStyle(m_wndSubresyncBar.GetBarStyle() | CBRS_TOOLTIPS | CBRS_FLYBY | CBRS_SIZE_DYNAMIC);
59.     m_wndSubresyncBar.EnableDocking(CBRS_ALIGN_ANY);
60.     m_wndSubresyncBar.SetHeight(200);
61.     m_dockingbars.AddTail(&m_wndSubresyncBar);
62.
63.     m_wndPlaylistBar.Create(this, AFX_IDW_DOCKBAR_BOTTOM);
64.     m_wndPlaylistBar.SetBarStyle(m_wndPlaylistBar.GetBarStyle() | CBRS_TOOLTIPS | CBRS_FLYBY | CBRS_SIZE_DYNAMIC);
65.     m_wndPlaylistBar.EnableDocking(CBRS_ALIGN_ANY);
66.     m_wndPlaylistBar.SetHeight(100);
67.     m_dockingbars.AddTail(&m_wndPlaylistBar);
68.     m_wndPlaylistBar.LoadPlaylist(GetRecentFile());
69.
70.     m_wndEditListEditor.Create(this, AFX_IDW_DOCKBAR_RIGHT);
71.     m_wndEditListEditor.SetBarStyle(m_wndEditListEditor.GetBarStyle() | CBRS_TOOLTIPS | CBRS_FLYBY | CBRS_SIZE_DYNAMIC);
72.     m_wndEditListEditor.EnableDocking(CBRS_ALIGN_ANY);
73.     m_dockingbars.AddTail(&m_wndEditListEditor);
74.     m_wndEditListEditor.SetHeight(100);
75.
76.     m_wndCaptureBar.Create(this, AFX_IDW_DOCKBAR_LEFT);
77.     m_wndCaptureBar.SetBarStyle(m_wndCaptureBar.GetBarStyle() | CBRS_TOOLTIPS | CBRS_FLYBY | CBRS_SIZE_DYNAMIC);
78.     m_wndCaptureBar.EnableDocking(CBRS_ALIGN_LEFT | CBRS_ALIGN_RIGHT);
79.     m_dockingbars.AddTail(&m_wndCaptureBar);
80.
81.     m_wndNavigationBar.Create(this, AFX_IDW_DOCKBAR_LEFT);
82.     m_wndNavigationBar.SetBarStyle(m_wndNavigationBar.GetBarStyle() | CBRS_TOOLTIPS | CBRS_FLYBY | CBRS_SIZE_DYNAMIC);
83.     m_wndNavigationBar.EnableDocking(CBRS_ALIGN_LEFT | CBRS_ALIGN_RIGHT);
84.     m_dockingbars.AddTail(&m_wndNavigationBar);
85.
86.     m_wndShaderEditorBar.Create(this, AFX_IDW_DOCKBAR_TOP);
87.     m_wndShaderEditorBar.SetBarStyle(m_wndShaderEditorBar.GetBarStyle() | CBRS_TOOLTIPS | CBRS_FLYBY | CBRS_SIZE_DYNAMIC);
88.     m_wndShaderEditorBar.EnableDocking(CBRS_ALIGN_ANY);
89.     m_dockingbars.AddTail(&m_wndShaderEditorBar);
90.
91.     // Hide all dockable bars by default
92.     POSITION pos = m_dockingbars.GetHeadPosition();
93.     while (pos) {
94.         m_dockingbars.GetNext(pos)->ShowWindow(SW_HIDE);
95.     }
96.
97.     m_fileDropTarget.Register(this);
98.
99.     const CAppSettings& s = AfxGetAppSettings();
100.
101.     // Load the controls
102.     m_nCS = s.nCS;
103.     ShowControls(m_nCS);
104.     //是否在最前?
105.     SetAlwaysOnTop(s.iOnTop);
106.     //显示系统托盘图标
107.     ShowTrayIcon(s.fTrayIcon);
108.     //焦点
109.     SetFocus();
110.     //创建CGraphThread类的线程
111.     m_pGraphThread = (CGraphThread*)AfxBeginThread(RUNTIME_CLASS(CGraphThread));
112.     //设置。。
113.     if (m_pGraphThread) {
114.         m_pGraphThread->SetMainFrame(this);
115.     }
116.
117.     if (s.nCmdLnWebServerPort != 0) {
118.         if (s.nCmdLnWebServerPort > 0) {
119.             StartWebServer(s.nCmdLnWebServerPort);
120.         } else if (s.fEnableWebServer) {
121.             StartWebServer(s.nWebServerPort);
122.         }
123.     }
124.
125.     // Casimir666 : reload Shaders
126.     {

```

```

127.         CString strList = s.strShaderList;
128.         CString strRes;
129.         int curPos = 0;
130.
131.         strRes = strList.Tokenize(_T("|"), curPos);
132.         while (!strRes.IsEmpty()) {
133.             m_shaderlabels.AddTail(strRes);
134.             strRes = strList.Tokenize(_T("|"), curPos);
135.         }
136.     }
137.     {
138.         CString strList = s.strShaderListScreenSpace;
139.         CString strRes;
140.         int curPos = 0;
141.
142.         strRes = strList.Tokenize(_T("|"), curPos);
143.         while (!strRes.IsEmpty()) {
144.             m_shaderlabelsScreenSpace.AddTail(strRes);
145.             strRes = strList.Tokenize(_T("|"), curPos);
146.         }
147.     }
148.
149.     m_bToggleShader = s.fToggleShader;
150.     m_bToggleShaderScreenSpace = s.fToggleShaderScreenSpace;
151.     //标题
152.     m_strTitle.LoadString(IDR_MAINFRAME);
153.
154. #ifdef MPCHC_LITE
155.     m_strTitle += _T(" Lite");
156. #endif
157.     //设置窗口标题
158.     SetWindowText(m_strTitle);
159.     m_Lcd.SetMediaTitle(LPCTSTR(m_strTitle));
160.
161.     WTSRegisterSessionNotification();
162.
163.     if (s.bNotifySkype) {
164.         m_pSkypeMoodMsgHandler.Attach(DEBUG_NEW SkypeMoodMsgHandler());
165.         m_pSkypeMoodMsgHandler->Connect(m_hWnd);
166.     }
167.
168.     return 0;
169. }</span>

```

在mpc-hc关闭的时候，会调用 OnDestroy()：

```

[cpp]
1. <span style="font-family:Arial;font-size:12px;">//关闭的时候加载
2. void CMainFrame::OnDestroy()
3. {
4.     WTSUnRegisterSessionNotification();
5.     //关闭系统托盘图标
6.     ShowTrayIcon(false);
7.     m_fileDropTarget.Revoke();
8.     //线程还在运行的话
9.     if (m_pGraphThread) {
10.         CAMMsgEvent e;
11.         //退出
12.         m_pGraphThread->PostThreadMessage(CGThread::TM_EXIT, 0, (LPARAM)&e);
13.         if (!e.Wait(5000)) {
14.             TRACE(_T("ERROR: Must call TerminateThread() on CMainFrame::m_pGraphThread->m_hThread\n"));
15.             TerminateThread(m_pGraphThread->m_hThread, (DWORD) - 1);
16.         }
17.     }
18.
19.     if (m_pFullscreenWnd) {
20.         if (m_pFullscreenWnd->IsWindow()) {
21.             m_pFullscreenWnd->DestroyWindow();
22.         }
23.         delete m_pFullscreenWnd;
24.     }
25.
26.     __super::OnDestroy();
27. }</span>

```

在关闭一个媒体的时候，会调用OnClose()：

```

1. //关闭的时候加载
2. void CMainFrame::OnClose()
3. {
4.     //获取设置
5.     CAppSettings& s = AfxGetAppSettings();
6.     // Casimir666 : save shaders list
7.     {
8.         POSITION pos;
9.         CString strList = "";
10.
11.         pos = m_shaderlabels.GetHeadPosition();
12.         while (pos) {
13.             strList += m_shaderlabels.GetAt(pos) + "|";
14.             m_dockingbars.GetNext(pos);
15.         }
16.         s.strShaderList = strList;
17.     }
18.     {
19.         POSITION pos;
20.         CString strList = "";
21.
22.         pos = m_shaderlabelsScreenSpace.GetHeadPosition();
23.         while (pos) {
24.             strList += m_shaderlabelsScreenSpace.GetAt(pos) + "|";
25.             m_dockingbars.GetNext(pos);
26.         }
27.         s.strShaderListScreenSpace = strList;
28.     }
29.
30.     s.fToggleShader = m_bToggleShader;
31.     s.fToggleShaderScreenSpace = m_bToggleShaderScreenSpace;
32.
33.     s.dZoomX = m_ZoomX;
34.     s.dZoomY = m_ZoomY;
35.     //存储播放列表
36.     m_wndPlaylistBar.SavePlaylist();
37.     //存储控制条
38.     SaveControlBars();
39.
40.     ShowWindow(SW_HIDE);
41.     //关闭媒体 (非private)
42.     CloseMedia();
43.
44.     s.WinLircClient.DisConnect();
45.     s.UIceClient.DisConnect();
46.
47.     SendAPICommand(CMD_DISCONNECT, L"\0"); // according to CMD_NOTIFYENDOFSTREAM (ctrl+f it here), you're not supposed to send NULL
48. re
49.     //调用父类onclose
50.     __super::OnClose();
51. }

```

同时还有一个定时器函数OnTimer()。根据不同的nIDEvent做不同的处理操作。

```

1. //定时刷新的操作
2. void CMainFrame::OnTimer(UINT_PTR nIDEvent)
3. {
4.     switch (nIDEvent) {
5.         //当前播放到的位置
6.         case TIMER_STREAMPOSPOLLER:
7.             if (m_iMediaLoadState == MLS_LOADED) {
8.                 REFERENCE_TIME rtNow = 0, rtDur = 0;
9.                 //播放方式是文件的时候 (还可是DVD或者摄像头)
10.                if (GetPlaybackMode() == PM_FILE) {
11.                    //当前位置
12.                    m_pMS->GetCurrentPosition(&rtNow);
13.                    //时常
14.                    m_pMS->GetDuration(&rtDur);
15.
16.                    // Casimir666 : autosave subtitle sync after play
17.                    if ((m_nCurSubtitle >= 0) && (m_rtCurSubPos != rtNow)) {
18.                        if (m_lSubtitleShift != 0) {
19.                            if (m_wndSubresyncBar.SaveToDisk()) {
20.                                m_OSD.DisplayMessage(OSD_TOPLEFT, ResStr(IDS_AG_SUBTITLES_SAVED), 500);
21.                            } else {
22.                                m_OSD.DisplayMessage(OSD_TOPLEFT, ResStr(IDS_MAINFRM_4));
23.                            }
24.                        }
25.                        m_nCurSubtitle = -1;
26.                        m_lSubtitleShift = 0;
27.                    }
28.
29.                    if (!m_fEndOfStream) {
30.                        CAppSettings& s = AfxGetAppSettings();
31.
32.                        if (m_bRememberFilePos) {

```

```

32.         if (m_bRememberFilePos) {
33.             FILE_POSITION* filePosition = s.filePositions.GetLatestEntry();
34.
35.             if (filePosition) {
36.                 filePosition->llPosition = rtNow;
37.
38.                 LARGE_INTEGER time;
39.                 QueryPerformanceCounter(&time);
40.                 LARGE_INTEGER freq;
41.                 QueryPerformanceFrequency(&freq);
42.                 if ((time.QuadPart - m_liLastSaveTime.QuadPart) >= 30 * freq.QuadPart) { // save every half of minute
43.
44.                     m_liLastSaveTime = time;
45.                     s.filePositions.SaveLatestEntry();
46.                 }
47.             }
48.         }
49.
50.         if (m_rtDurationOverride >= 0) {
51.             rtDur = m_rtDurationOverride;
52.         }
53.         //设置滑动条控件的参数 (位置等。。。)
54.         g_bNoDuration = rtDur <= 0;
55.         m_wndSeekBar.Enable(rtDur > 0);
56.         m_wndSeekBar.SetRange(0, rtDur);
57.         m_wndSeekBar.SetPos(rtNow);
58.         m_OSD.SetRange(0, rtDur);
59.         m_OSD.SetPos(rtNow);
60.         m_Lcd.SetMediaRange(0, rtDur);
61.         m_Lcd.SetMediaPos(rtNow);
62.     } else if (GetPlaybackMode() == PM_CAPTURE) {
63.         //如果是摄像头的话,就没有时长信息了
64.         m_pMS->GetCurrentPosition(&rtNow);
65.         if (m_fCapturing && m_wndCaptureBar.m_capdlg.m_pMux) {
66.             CComQIPtr<IMediaSeeking> pMuxMS = m_wndCaptureBar.m_capdlg.m_pMux;
67.             if (!pMuxMS || FAILED(pMuxMS->GetCurrentPosition(&rtNow))) {
68.                 rtNow = 0;
69.             }
70.         }
71.
72.         if (m_rtDurationOverride >= 0) {
73.             rtDur = m_rtDurationOverride;
74.         }
75.
76.         g_bNoDuration = rtDur <= 0;
77.         m_wndSeekBar.Enable(false);
78.         m_wndSeekBar.SetRange(0, rtDur);
79.         m_wndSeekBar.SetPos(rtNow);
80.         m_OSD.SetRange(0, rtDur);
81.         m_OSD.SetPos(rtNow);
82.         m_Lcd.SetMediaRange(0, rtDur);
83.         m_Lcd.SetMediaPos(rtNow);
84.     }
85.
86.     if (m_pCAP && GetPlaybackMode() != PM_FILE) {
87.         g_bExternalSubtitleTime = true;
88.         if (m_pDVDI) {
89.             DVD_PLAYBACK_LOCATION2 Location;
90.             if (m_pDVDI->GetCurrentLocation(&Location) == S_OK) {
91.                 double fps = Location.TimeCodeFlags == DVD_TC_FLAG_25fps ? 25.0
92.                     : Location.TimeCodeFlags == DVD_TC_FLAG_30fps ? 30.0
93.                     : Location.TimeCodeFlags == DVD_TC_FLAG_DropFrame ? 29.97
94.                     : 25.0;
95.
96.                 REFERENCE_TIME rtTimeCode = HMSF2RT(Location.TimeCode, fps);
97.                 m_pCAP->SetTime(rtTimeCode);
98.             } else {
99.                 m_pCAP->SetTime(*rtNow/m_wndSeekBar.GetPos());
100.            }
101.        } else {
102.            // Set rtNow to support DVB subtitle
103.            m_pCAP->SetTime(rtNow);
104.        }
105.    } else {
106.        g_bExternalSubtitleTime = false;
107.    }
108. }
109. break;
110. case TIMER_STREAMPOSPOLLER2:
111.     if (m_iMediaLoadState == MLS_LOADED) {
112.         __int64 start, stop, pos;
113.         m_wndSeekBar.GetRange(start, stop);
114.         pos = m_wndSeekBar.GetPosReal();
115.
116.         GUID tf;
117.         m_pMS->GetTimeFormat(&tf);
118.
119.         if (GetPlaybackMode() == PM_CAPTURE && !m_fCapturing) {
120.             CString str = ResStr(IDS_CAPTURE_LIVE);
121.
122.             long lChannel = 0, lVivSub = 0, lAudSub = 0;

```

```

123.         long channel = 0, vivsub = 0, audsub = 0;
124.         if (m_pAMTuner
125.             && m_wndCaptureBar.m_capdlg.IsTunerActive()
126.             && SUCCEEDED(m_pAMTuner->get_Channel(&lChannel, &lVivSub, &lAudSub))) {
127.             CString ch;
128.             ch.Format(_T(" (ch%d)", lChannel);
129.             str += ch;
130.         }
131.         m_wndStatusBar.SetStatusTimer(str);
132.     } else {
133.         m_wndStatusBar.SetStatusTimer(pos, stop, !!m_wndSubresyncBar.IsWindowVisible(), &tft);
134.         if (m_bRemainingTime) {
135.             m OSD.DisplayMessage(OSD_TOPLEFT, m_wndStatusBar.GetStatusTimer());
136.         }
137.     }
138.
139.     m_wndSubresyncBar.SetTime(pos);
140.
141.     if (m_pCAP && GetMediaState() == State_Paused) {
142.         m_pCAP->Paint(false);
143.     }
144. }
145. break;
146. case TIMER_FULLSCREENCONTROLBARHIDER: {
147.     CPoint p;
148.     GetCursorPos(&p);
149.
150.     CRect r;
151.     GetWindowRect(r);
152.     bool fCursorOutside = !r.PtInRect(p);
153.
154.     CWnd* pWnd = WindowFromPoint(p);
155.     if (pWnd && (m_wndView == *pWnd || m_wndView.IsChild(pWnd) || fCursorOutside)) {
156.         if (AfxGetAppSettings().nShowBarsWhenFullScreenTimeOut >= 0) {
157.             ShowControls(CS_NONE);
158.         }
159.     }
160. }
161. break;
162. case TIMER_FULLSCREENMOUSEHIDER: {
163.     CPoint p;
164.     GetCursorPos(&p);
165.
166.     CRect r;
167.     GetWindowRect(r);
168.     bool fCursorOutside = !r.PtInRect(p);
169.     CWnd* pWnd = WindowFromPoint(p);
170.     if (IsD3DFullScreenMode()) {
171.         if (pWnd && !m_bInOptions && *pWnd == *m_pFullscreenWnd) {
172.             m_pFullscreenWnd->ShowCursor(false);
173.         }
174.         KillTimer(TIMER_FULLSCREENMOUSEHIDER);
175.     } else {
176.         if (pWnd && !m_bInOptions && (m_wndView == *pWnd || m_wndView.IsChild(pWnd) || fCursorOutside)) {
177.             m_fHideCursor = true;
178.             SetCursor(nullptr);
179.         }
180.     }
181. }
182. break;
183. //统计量
184. case TIMER_STATS: {
185.     //接收端质量信息：抖动，抖动，视音频同步情况等。。。
186.     if (m_pQP) {
187.         CString rate;
188.         rate.Format(_T("%.2f"), m_dSpeedRate);
189.         rate = _T("(") + rate + _T("x)");
190.         //信息
191.         CString info;
192.         int val = 0;
193.         //平均帧率
194.         m_pQP->get_AvgFrameRate(&val); // We hang here due to a lock that never gets released.
195.         info.Format(_T("%.02d %s"), val / 100, val % 100, rate);
196.         m_wndStatsBar.SetLine(ResStr(IDS_AG_FRAMERATE), info);
197.
198.         int avg, dev;
199.         //抖动
200.         m_pQP->get_AvgSyncOffset(&avg);
201.         m_pQP->get_DevSyncOffset(&dev);
202.         info.Format(ResStr(IDS_STATSBAR_SYNC_OFFSET_FORMAT), avg, dev);
203.         m_wndStatsBar.SetLine(ResStr(IDS_STATSBAR_SYNC_OFFSET), info);
204.         //掉帧
205.         int drawn, dropped;
206.         m_pQP->get_FramesDrawn(&drawn);
207.         m_pQP->get_FramesDroppedInRenderer(&dropped);
208.         info.Format(IDS_MAINFRM_6, drawn, dropped);
209.         m_wndStatsBar.SetLine(ResStr(IDS_AG_FRAMES), info);
210.         //抖动
211.         m_pQP->get_Jitter(&val);
212.         info.Format(_T("%d ms"), val);
213.         m_wndStatsBar.SetLine(ResStr(IDS_STATSBAR_JITTER), info);

```

```

214.     }
215.     //缓存信息
216.     if (m_pBI) {
217.         CATList<CString> sl;
218.         //获取数量
219.         for (int i = 0, j = m_pBI->GetCount(); i < j; i++) {
220.             int samples, size;
221.             //获取缓存状态
222.             if (S_OK == m_pBI->GetStatus(i, samples, size)) {
223.                 CString str;
224.                 str.Format(_T("[%d]: %03d/%d KB"), i, samples, size / 1024);
225.                 sl.AddTail(str);
226.             }
227.         }
228.
229.         if (!sl.IsEmpty()) {
230.             CString str;
231.             str.Format(_T("%s (p%u)"), Implode(sl, ' '), m_pBI->GetPriority());
232.
233.             m_wndStatsBar.SetLine(ResStr(IDS_AG_BUFFERS), str);
234.         }
235.     }
236.     //比特率信息
237.     CInterfaceList<IBitRateInfo> pBRIs;
238.
239.     BeginEnumFilters(m_pGB, pEF, pBF) {
240.         BeginEnumPins(pBF, pEP, pPin) {
241.             if (CComQIPtr<IBitRateInfo> pBRI = pPin) {
242.                 pBRIs.AddTail(pBRI);
243.             }
244.         }
245.     EndEnumPins;
246.
247.     if (!pBRIs.IsEmpty()) {
248.         CATList<CString> sl;
249.
250.         POSITION pos = pBRIs.GetHeadPosition();
251.         for (int i = 0; pos; i++) {
252.             //比特率接口
253.             IBitRateInfo* pBRI = pBRIs.GetNext(pos);
254.             //当前比特率
255.             DWORD cur = pBRI->GetCurrentBitRate() / 1000;
256.             //平均比特率
257.             DWORD avg = pBRI->GetAverageBitRate() / 1000;
258.
259.             if (avg == 0) {
260.                 continue;
261.             }
262.             //添加到字符串
263.             CString str;
264.             if (cur != avg) {
265.                 str.Format(_T("[%d]: %u/%u Kb/s"), i, avg, cur);
266.             } else {
267.                 str.Format(_T("[%d]: %u Kb/s"), i, avg);
268.             }
269.             //加入
270.             sl.AddTail(str);
271.         }
272.
273.         if (!sl.IsEmpty()) {
274.             m_wndStatsBar.SetLine(ResStr(IDS_STATSBAR_BITRATE), Implode(sl, ' ') + ResStr(IDS_STATSBAR_BITRATE_AVG_CUR));
275.         }
276.
277.         break;
278.     }
279. }
280. EndEnumFilters;
281.
282. if (GetPlaybackMode() == PM_DVD) { // we also use this timer to update the info panel for DVD playback
283.     ULONG ulAvailable, ulCurrent;
284.
285.     // Location
286.
287.     CString Location('-');
288.
289.     DVD_PLAYBACK_LOCATION2 loc;
290.     ULONG ulNumOfVolumes, ulVolume;
291.     DVD_DISC_SIDE Side;
292.     ULONG ulNumOfTitles;
293.     ULONG ulNumOfChapters;
294.
295.     if (SUCCEEDED(m_pDVDI->GetCurrentLocation(&loc))
296.         && SUCCEEDED(m_pDVDI->GetNumberOfChapters(loc.TitleNum, &ulNumOfChapters))
297.         && SUCCEEDED(m_pDVDI->GetDVDVolumeInfo(&ulNumOfVolumes, &ulVolume, &Side, &ulNumOfTitles))) {
298.         Location.Format(IDS_MAINFRM_9,
299.             ulVolume, ulNumOfVolumes,
300.             loc.TitleNum, ulNumOfTitles,
301.             loc.ChapterNum, ulNumOfChapters);
302.         ULONG tsec = (loc.TimeCode.bHours * 3600)
303.             + (loc.TimeCode.bMinutes * 60)

```

```

304.         + (loc.TimeCode.bSeconds);
305.         /* This might not always work, such as on resume */
306.         if (loc.ChapterNum != m_lCurrentChapter) {
307.             m_lCurrentChapter = loc.ChapterNum;
308.             m_lChapterStartTime = tsec;
309.         } else {
310.             /* If a resume point was used, and the user chapter jumps,
311.              then it might do some funky time jumping. Try to 'fix' the
312.              chapter start time if this happens */
313.             if (m_lChapterStartTime > tsec) {
314.                 m_lChapterStartTime = tsec;
315.             }
316.         }
317.     }
318.
319.     m_wndInfoBar.SetLine(ResStr(IDS_INFOBAR_LOCATION), Location);
320.
321.     // Video
322.
323.     CString Video('-');
324.
325.     DVD_VideoAttributes VATR;
326.
327.     if (SUCCEEDED(m_pDVDI->GetCurrentAngle(&ulAvailable, &ulCurrent))
328.         && SUCCEEDED(m_pDVDI->GetCurrentVideoAttributes(&VATR))) {
329.         Video.Format(IDS_MAINFRM_10,
330.                     ulCurrent, ulAvailable,
331.                     VATR.ulSourceResolutionX, VATR.ulSourceResolutionY, VATR.ulFrameRate,
332.                     VATR.ulAspectX, VATR.ulAspectY);
333.     }
334.
335.     m_wndInfoBar.SetLine(ResStr(IDS_INFOBAR_VIDEO), Video);
336.
337.     // Audio
338.
339.     CString Audio('-');
340.
341.     DVD_AudioAttributes AATR;
342.
343.     if (SUCCEEDED(m_pDVDI->GetCurrentAudio(&ulAvailable, &ulCurrent))
344.         && SUCCEEDED(m_pDVDI->GetAudioAttributes(ulCurrent, &AATR))) {
345.         CString lang;
346.         if (AATR.Language) {
347.             int len = GetLocaleInfo(AATR.Language, LOCALE_SENGLANGUAGE, lang.GetBuffer(64), 64);
348.             lang.ReleaseBufferSetLength(max(len - 1, 0));
349.         } else {
350.             lang.Format(IDS_AG_UNKNOWN, ulCurrent + 1);
351.         }
352.
353.         switch (AATR.LanguageExtension) {
354.             case DVD_AUD_EXT_NotSpecified:
355.                 default:
356.                     break;
357.             case DVD_AUD_EXT_Captions:
358.                 lang += _T(" (Captions)");
359.                 break;
360.             case DVD_AUD_EXT_VisuallyImpaired:
361.                 lang += _T(" (Visually Impaired)");
362.                 break;
363.             case DVD_AUD_EXT_DirectorComments1:
364.                 lang += _T(" (Director Comments 1)");
365.                 break;
366.             case DVD_AUD_EXT_DirectorComments2:
367.                 lang += _T(" (Director Comments 2)");
368.                 break;
369.         }
370.
371.         CString format = GetDVDAudioFormatName(AATR);
372.
373.         Audio.Format(IDS_MAINFRM_11,
374.                     lang,
375.                     format,
376.                     AATR.dwFrequency,
377.                     AATR.bQuantization,
378.                     AATR.bNumberOfChannels,
379.                     (AATR.bNumberOfChannels > 1 ? ResStr(IDS_MAINFRM_13) : ResStr(IDS_MAINFRM_12)));
380.
381.         m_wndStatusBar.SetStatusBitmap(
382.             AATR.bNumberOfChannels == 1 ? IDB_AUDIOTYPE_MONO
383.             : AATR.bNumberOfChannels >= 2 ? IDB_AUDIOTYPE_STEREO
384.             : IDB_AUDIOTYPE_NOAUDIO);
385.     }
386.
387.     m_wndInfoBar.SetLine(ResStr(IDS_INFOBAR_AUDIO), Audio);
388.
389.     // Subtitles
390.
391.     CString Subtitles('-');
392.
393.     BOOL bIsDisabled;
394.     DVD_SubpictureAttributes SATR;

```



```

395.
396.         if (SUCCEEDED(m_pDVDI->GetCurrentSubpicture(&ulAvailable, &ulCurrent, &bIsDisabled))
397.             && SUCCEEDED(m_pDVDI->GetSubpictureAttributes(ulCurrent, &SATR))) {
398.             CString lang;
399.             int len = GetLocaleInfo(SATR.Language, LOCALE_SENGLANGUAGE, lang.GetBuffer(64), 64);
400.             lang.ReleaseBufferSetLength(max(len - 1, 0));
401.
402.             switch (SATR.LanguageExtension) {
403.             case DVD_SP_EXT_NotSpecified:
404.             default:
405.                 break;
406.             case DVD_SP_EXT_Caption_Normal:
407.                 lang += _T("");
408.                 break;
409.             case DVD_SP_EXT_Caption_Big:
410.                 lang += _T(" (Big)");
411.                 break;
412.             case DVD_SP_EXT_Caption_Children:
413.                 lang += _T(" (Children)");
414.                 break;
415.             case DVD_SP_EXT_CC_Normal:
416.                 lang += _T(" (CC)");
417.                 break;
418.             case DVD_SP_EXT_CC_Big:
419.                 lang += _T(" (CC Big)");
420.                 break;
421.             case DVD_SP_EXT_CC_Children:
422.                 lang += _T(" (CC Children)");
423.                 break;
424.             case DVD_SP_EXT_Forced:
425.                 lang += _T(" (Forced)");
426.                 break;
427.             case DVD_SP_EXT_DirectorComments_Normal:
428.                 lang += _T(" (Director Comments)");
429.                 break;
430.             case DVD_SP_EXT_DirectorComments_Big:
431.                 lang += _T(" (Director Comments, Big)");
432.                 break;
433.             case DVD_SP_EXT_DirectorComments_Children:
434.                 lang += _T(" (Director Comments, Children)");
435.                 break;
436.             }
437.
438.             if (bIsDisabled) {
439.                 lang = _T("-");
440.             }
441.
442.             Subtitles.Format(_T("%s"),
443.                             lang);
444.         }
445.
446.         m_wndInfoBar.SetLine(ResStr(IDS_INFOBAR_SUBTITLES), Subtitles);
447.     } else if (GetPlaybackMode() == PM_CAPTURE && AfxGetAppSettings().iDefaultCaptureDevice == 1) {
448.         CComQIPtr<IBDATuner> pTun = m_pGB;
449.         BOOLEAN bPresent;
450.         BOOLEAN bLocked;
451.         LONG lDbStrength;
452.         LONG lPercentQuality;
453.         CString Signal;
454.
455.         if (SUCCEEDED(pTun->GetStats(bPresent, bLocked, lDbStrength, lPercentQuality)) && bPresent) {
456.             Signal.Format(ResStr(IDS_STATSBAR_SIGNAL_FORMAT), (int)lDbStrength, lPercentQuality);
457.             m_wndStatsBar.SetLine(ResStr(IDS_STATSBAR_SIGNAL), Signal);
458.         }
459.     } else if (GetPlaybackMode() == PM_FILE) {
460.         UpdateChapterInInfoBar();
461.     }
462.
463.     if (GetMediaState() == State_Running && !m_fAudioOnly) {
464.         BOOL fActive = FALSE;
465.         if (SystemParametersInfo(SPI_GETSCREENSAVEACTIVE, 0, &fActive, 0)) {
466.             SystemParametersInfo(SPI_SETSCREENSAVEACTIVE, FALSE, nullptr, SPIF_SENDWININICHANGE); // this might not be
needed at all...
467.             SystemParametersInfo(SPI_SETSCREENSAVEACTIVE, fActive, nullptr, SPIF_SENDWININICHANGE);
468.         }
469.
470.         fActive = FALSE;
471.         if (SystemParametersInfo(SPI_GETPOWEROFFACTIVE, 0, &fActive, 0)) {
472.             SystemParametersInfo(SPI_SETPOWEROFFACTIVE, FALSE, nullptr, SPIF_SENDWININICHANGE); // this might not be ne
eded at all...
473.             SystemParametersInfo(SPI_SETPOWEROFFACTIVE, fActive, nullptr, SPIF_SENDWININICHANGE);
474.         }
475.         // prevent screensaver activate, monitor sleep/turn off after playback
476.         SetThreadExecutionState(ES_SYSTEM_REQUIRED | ES_DISPLAY_REQUIRED);
477.     }
478. }
479. break;
480. case TIMER_STATUSERASER: {
481.     KillTimer(TIMER_STATUSERASER);
482.     m_playingmsg.Empty();
483. }

```

```
484.         break;
485.     case TIMER_DVBINFO_UPDATER:
486.         KillTimer(TIMER_DVBINFO_UPDATER);
487.         ShowCurrentChannelInfo(false, false);
488.         break;
489.     }
490.
491.     __super::OnTimer(nIDEvent);
492. }
```

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/13298397>

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