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：5865

==========

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系的功能。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
<span style="font-family:Arial;font-size:12px;">//刚打开的时候加载
int CMainFrame::OnCreate(LPCREATESTRUCT lpCreateStruct)
{
    if (__super::OnCreate(lpCreateStruct) == -1) {
        return -1;
    }
    //加载菜单
    m_popup.LoadMenu(IDR_POPUP);
    m_popupmain.LoadMenu(IDR_POPUPMAIN);

    // create a view to occupy the client area of the frame
    // 创建视频画面部分？
    if (!m_wndView.Create(nullptr, nullptr, AFX_WS_DEFAULT_VIEW,
                          CRect(0, 0, 0, 0), this, AFX_IDW_PANE_FIRST, nullptr)) {
        TRACE(_T("Failed to create view window\n"));
        return -1;
    }
    // Should never be RTLed
    m_wndView.ModifyStyleEx(WS_EX_LAYOUTRTL, WS_EX_NOINHERITLAYOUT);

    // static bars
    //各种状态栏
    BOOL bResult = m_wndStatusBar.Create(this);
    if (bResult) {
        bResult = m_wndStatsBar.Create(this);
    }
    if (bResult) {
        bResult = m_wndInfoBar.Create(this);
    }
    if (bResult) {
        bResult = m_wndToolBar.Create(this);
    }
    if (bResult) {
        bResult = m_wndSeekBar.Create(this);
    }
```

```
    if (!bResult) {
        TRACE(_T("Failed to create all control bars\n"));
        return -1;      // fail to create
    }
    // 各种Bar
    m_pFullscreenWnd = DEBUG_NEW CFullscreenWnd(this);

    m_bars.AddTail(&m_wndSeekBar);
    m_bars.AddTail(&m_wndToolBar);
    m_bars.AddTail(&m_wndInfoBar);
    m_bars.AddTail(&m_wndStatsBar);
    m_bars.AddTail(&m_wndStatusBar);

    m_wndSeekBar.Enable(false);

    // dockable bars
    // 可停靠的
    EnableDocking(CBRS_ALIGN_ANY);

    m_dockingbars.RemoveAll();

    m_wndSubresyncBar.Create(this, AFX_IDW_DOCKBAR_TOP, &m_csSubLock);
    m_wndSubresyncBar.SetBarStyle(m_wndSubresyncBar.GetBarStyle() | CBRS_TOOLTIPS | CBRS_FLYBY | CBRS_SIZE_DYNAMIC);
    m_wndSubresyncBar.EnableDocking(CBRS_ALIGN_ANY);
    m_wndSubresyncBar.SetHeight(200);
    m_dockingbars.AddTail(&m_wndSubresyncBar);

    m_wndPlaylistBar.Create(this, AFX_IDW_DOCKBAR_BOTTOM);
    m_wndPlaylistBar.SetBarStyle(m_wndPlaylistBar.GetBarStyle() | CBRS_TOOLTIPS | CBRS_FLYBY | CBRS_SIZE_DYNAMIC);
    m_wndPlaylistBar.EnableDocking(CBRS_ALIGN_ANY);
    m_wndPlaylistBar.SetHeight(100);
    m_dockingbars.AddTail(&m_wndPlaylistBar);
    m_wndPlaylistBar.LoadPlaylist(GetRecentFile());

    m_wndEditListEditor.Create(this, AFX_IDW_DOCKBAR_RIGHT);
    m_wndEditListEditor.SetBarStyle(m_wndEditListEditor.GetBarStyle() | CBRS_TOOLTIPS | CBRS_FLYBY | CBRS_SIZE_DYNAMIC);
    m_wndEditListEditor.EnableDocking(CBRS_ALIGN_ANY);
    m_dockingbars.AddTail(&m_wndEditListEditor);
    m_wndEditListEditor.SetHeight(100);

    m_wndCaptureBar.Create(this, AFX_IDW_DOCKBAR_LEFT);
    m_wndCaptureBar.SetBarStyle(m_wndCaptureBar.GetBarStyle() | CBRS_TOOLTIPS | CBRS_FLYBY | CBRS_SIZE_DYNAMIC);
    m_wndCaptureBar.EnableDocking(CBRS_ALIGN_LEFT | CBRS_ALIGN_RIGHT);
    m_dockingbars.AddTail(&m_wndCaptureBar);

    m_wndNavigationBar.Create(this, AFX_IDW_DOCKBAR_LEFT);
    m_wndNavigationBar.SetBarStyle(m_wndNavigationBar.GetBarStyle() | CBRS_TOOLTIPS | CBRS_FLYBY | CBRS_SIZE_DYNAMIC);
    m_wndNavigationBar.EnableDocking(CBRS_ALIGN_LEFT | CBRS_ALIGN_RIGHT);
    m_dockingbars.AddTail(&m_wndNavigationBar);

    m_wndShaderEditorBar.Create(this, AFX_IDW_DOCKBAR_TOP);
    m_wndShaderEditorBar.SetBarStyle(m_wndShaderEditorBar.GetBarStyle() | CBRS_TOOLTIPS | CBRS_FLYBY | CBRS_SIZE_DYNAMIC);
    m_wndShaderEditorBar.EnableDocking(CBRS_ALIGN_ANY);
    m_dockingbars.AddTail(&m_wndShaderEditorBar);

    // Hide all dockable bars by default
    POSITION pos = m_dockingbars.GetHeadPosition();
    while (pos) {
        m_dockingbars.GetNext(pos)->ShowWindow(SW_HIDE);
    }

    m_fileDropTarget.Register(this);

    const CAppSettings& s = AfxGetAppSettings();

    // Load the controls
    m_nCS = s.nCS;
    ShowControls(m_nCS);
    //是否在最前？
    SetAlwaysOnTop(s.iOnTop);
    //显示系统托盘图标
    ShowTrayIcon(s.fTrayIcon);
    //焦点
    SetFocus();
    //创建CGraphThread类的线程
    m_pGraphThread = (CGraphThread*)AfxBeginThread(RUNTIME_CLASS(CGraphThread));
    //设置。。
    if (m_pGraphThread) {
        m_pGraphThread->SetMainFrame(this);
    }

    if (s.nCmdlnWebServerPort != 0) {
        if (s.nCmdlnWebServerPort > 0) {
            StartWebServer(s.nCmdlnWebServerPort);
        } else if (s.fEnableWebServer) {
            StartWebServer(s.nWebServerPort);
        }
    }

    // Casimir666 : reload Shaders
    {
```

```
        CString strList = s.strShaderList;
        CString strRes;
        int curPos = 0;

        strRes = strList.Tokenize(_T("|"), curPos);
        while (!strRes.IsEmpty()) {
            m_shaderlabels.AddTail(strRes);
            strRes = strList.Tokenize(_T("|"), curPos);
        }
    }
    {
        CString strList = s.strShaderListScreenSpace;
        CString strRes;
        int curPos = 0;

        strRes = strList.Tokenize(_T("|"), curPos);
        while (!strRes.IsEmpty()) {
            m_shaderlabelsScreenSpace.AddTail(strRes);
            strRes = strList.Tokenize(_T("|"), curPos);
        }
    }

    m_bToggleShader = s.fToggleShader;
    m_bToggleShaderScreenSpace = s.fToggleShaderScreenSpace;
    //标题
    m_strTitle.LoadString(IDR_MAINFRAME);

#ifdef MPCHC_LITE
    m_strTitle += _T(" Lite");
#endif
    //设置窗口标题
    SetWindowText(m_strTitle);
    m_Lcd.SetMediaTitle(LPCTSTR(m_strTitle));

    WTSRegisterSessionNotification();

    if (s.bNotifySkype) {
        m_pSkypeMoodMsgHandler.Attach(DEBUG_NEW SkypeMoodMsgHandler());
        m_pSkypeMoodMsgHandler->Connect(m_hWnd);
    }

    return 0;
}</span>
```

在mpc-hc关闭的时候，会调用 OnDestroy()：

```cpp
<span style="font-family:Arial;font-size:12px;">//关闭的时候加载
void CMainFrame::OnDestroy()
{
    WTSUnRegisterSessionNotification();
    //关闭系统托盘图标
    ShowTrayIcon(false);
    m_fileDropTarget.Revoke();
    //线程还在运行的话
    if (m_pGraphThread) {
        CAMMsgEvent e;
        //退出
        m_pGraphThread->PostThreadMessage(CGraphThread::TM_EXIT, 0, (LPARAM)&e);
        if (!e.Wait(5000)) {
            TRACE(_T("ERROR: Must call TerminateThread() on CMainFrame::m_pGraphThread->m_hThread\n"));
            TerminateThread(m_pGraphThread->m_hThread, (DWORD) - 1);
        }
    }

    if (m_pFullscreenWnd) {
        if (m_pFullscreenWnd->IsWindow()) {
            m_pFullscreenWnd->DestroyWindow();
        }
        delete m_pFullscreenWnd;
    }

    __super::OnDestroy();
}</span>
```

在关闭一个媒体的时候，会调用OnClose()：

```cpp
//关闭的时候加载
void CMainFrame::OnClose()
{
    //获取设置
    CAppSettings& s = AfxGetAppSettings();
    // Casimir666 : save shaders list
    {
        POSITION pos;
        CString strList = "";

        pos = m_shaderlabels.GetHeadPosition();
        while (pos) {
            strList += m_shaderlabels.GetAt(pos) + "|";
            m_dockingbars.GetNext(pos);
        }
        s.strShaderList = strList;
    }
    {
        POSITION pos;
        CString  strList = "";

        pos = m_shaderlabelsScreenSpace.GetHeadPosition();
        while (pos) {
            strList += m_shaderlabelsScreenSpace.GetAt(pos) + "|";
            m_dockingbars.GetNext(pos);
        }
        s.strShaderListScreenSpace = strList;
    }

    s.fToggleShader = m_bToggleShader;
    s.fToggleShaderScreenSpace = m_bToggleShaderScreenSpace;

    s.dZoomX = m_ZoomX;
    s.dZoomY = m_ZoomY;
    //存储播放列表
    m_wndPlaylistBar.SavePlaylist();
    //存储控制条
    SaveControlBars();

    ShowWindow(SW_HIDE);
    //关闭媒体（非private）
    CloseMedia();

    s.WinLircClient.DisConnect();
    s.UIceClient.DisConnect();

    SendAPICommand(CMD_DISCONNECT, L"\0");  // according to CMD_NOTIFYENDOFSTREAM (ctrl+f it here), you're not supposed to send NULL re
    //调用父类onclose
    __super::OnClose();
}
```

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。

```cpp
//定时刷新的操作
void CMainFrame::OnTimer(UINT_PTR nIDEvent)
{
    switch (nIDEvent) {
        //当前播放到的位置
        case TIMER_STREAMPOSPOLLER:
            if (m_iMediaLoadState == MLS_LOADED) {
                REFERENCE_TIME rtNow = 0, rtDur = 0;
                //播放方式是文件的时候（还可以是DVD或者摄像头）
                if (GetPlaybackMode() == PM_FILE) {
                    //当前位置
                    m_pMS->GetCurrentPosition(&rtNow);
                    //时常
                    m_pMS->GetDuration(&rtDur);

                    // Casimir666 : autosave subtitle sync after play
                    if ((m_nCurSubtitle >= 0) && (m_rtCurSubPos != rtNow)) {
                        if (m_lSubtitleShift != 0) {
                            if (m_wndSubresyncBar.SaveToDisk()) {
                                m_OSD.DisplayMessage(OSD_TOPLEFT, ResStr(IDS_AG_SUBTITLES_SAVED), 500);
                            } else {
                                m_OSD.DisplayMessage(OSD_TOPLEFT, ResStr(IDS_MAINFRM_4));
                            }
                        }
                        m_nCurSubtitle   = -1;
                        m_lSubtitleShift = 0;
                    }

                    if (!m_fEndOfStream) {
                        CAppSettings& s = AfxGetAppSettings();

                        if (m_bRememberFilePos) {
```

```
                    if (m_bRememberFilePos) {
                        FILE_POSITION* filePosition = s.filePositions.GetLatestEntry();

                        if (filePosition) {
                            filePosition->llPosition = rtNow;

                            LARGE_INTEGER time;
                            QueryPerformanceCounter(&time);
                            LARGE_INTEGER freq;
                            QueryPerformanceFrequency(&freq);
                            if ((time.QuadPart - m_liLastSaveTime.QuadPart) >= 30 * freq.QuadPart) { // save every half of minute

                                m_liLastSaveTime = time;
                                s.filePositions.SaveLatestEntry();
                            }
                        }
                    }
                }

                if (m_rtDurationOverride >= 0) {
                    rtDur = m_rtDurationOverride;
                }
                //设置滑动条控件的参数（位置等。。。）
                g_bNoDuration = rtDur <= 0;
                m_wndSeekBar.Enable(rtDur > 0);
                m_wndSeekBar.SetRange(0, rtDur);
                m_wndSeekBar.SetPos(rtNow);
                m_OSD.SetRange(0, rtDur);
                m_OSD.SetPos(rtNow);
                m_Lcd.SetMediaRange(0, rtDur);
                m_Lcd.SetMediaPos(rtNow);
            } else if (GetPlaybackMode() == PM_CAPTURE) {
                //如果是摄像头的话，就没有时长信息了
                m_pMS->GetCurrentPosition(&rtNow);
                if (m_fCapturing && m_wndCaptureBar.m_capdlg.m_pMux) {
                    CComQIPtr<IMediaSeeking> pMuxMS = m_wndCaptureBar.m_capdlg.m_pMux;
                    if (!pMuxMS || FAILED(pMuxMS->GetCurrentPosition(&rtNow))) {
                        rtNow = 0;
                    }
                }

                if (m_rtDurationOverride >= 0) {
                    rtDur = m_rtDurationOverride;
                }

                g_bNoDuration = rtDur <= 0;
                m_wndSeekBar.Enable(false);
                m_wndSeekBar.SetRange(0, rtDur);
                m_wndSeekBar.SetPos(rtNow);
                m_OSD.SetRange(0, rtDur);
                m_OSD.SetPos(rtNow);
                m_Lcd.SetMediaRange(0, rtDur);
                m_Lcd.SetMediaPos(rtNow);
            }

            if (m_pCAP && GetPlaybackMode() != PM_FILE) {
                g_bExternalSubtitleTime = true;
                if (m_pDVDI) {
                    DVD_PLAYBACK_LOCATION2 Location;
                    if (m_pDVDI->GetCurrentLocation(&Location) == S_OK) {
                        double fps = Location.TimeCodeFlags == DVD_TC_FLAG_25fps ? 25.0
                                   : Location.TimeCodeFlags == DVD_TC_FLAG_30fps ? 30.0
                                   : Location.TimeCodeFlags == DVD_TC_FLAG_DropFrame ? 29.97
                                   : 25.0;

                        REFERENCE_TIME rtTimeCode = HMSF2RT(Location.TimeCode, fps);
                        m_pCAP->SetTime(rtTimeCode);
                    } else {
                        m_pCAP->SetTime(/*rtNow*/m_wndSeekBar.GetPos());
                    }
                } else {
                    // Set rtNow to support DVB subtitle
                    m_pCAP->SetTime(rtNow);
                }
            } else {
                g_bExternalSubtitleTime = false;
            }
        }
        break;
    case TIMER_STREAMPOSPOLLER2:
        if (m_iMediaLoadState == MLS_LOADED) {
            __int64 start, stop, pos;
            m_wndSeekBar.GetRange(start, stop);
            pos = m_wndSeekBar.GetPosReal();

            GUID tf;
            m_pMS->GetTimeFormat(&tf);

            if (GetPlaybackMode() == PM_CAPTURE && !m_fCapturing) {
                CString str = ResStr(IDS_CAPTURE_LIVE);

                long lChannel = 0, lVivSub = 0, lAudSub = 0;
```

```
                    long lChannel = 0, lVivSub = 0, lAudSub = 0;
                    if (m_pAMTuner
                            && m_wndCaptureBar.m_capdlg.IsTunerActive()
                            && SUCCEEDED(m_pAMTuner->get_Channel(&lChannel, &lVivSub, &lAudSub))) {
                        CString ch;
                        ch.Format(_T(" (ch%d)"), lChannel);
                        str += ch;
                    }

                    m_wndStatusBar.SetStatusTimer(str);
                } else {
                    m_wndStatusBar.SetStatusTimer(pos, stop, !!m_wndSubresyncBar.IsWindowVisible(), &tf);
                    if (m_bRemainingTime) {
                        m_OSD.DisplayMessage(OSD_TOPLEFT, m_wndStatusBar.GetStatusTimer());
                    }
                }

                m_wndSubresyncBar.SetTime(pos);

                if (m_pCAP && GetMediaState() == State_Paused) {
                    m_pCAP->Paint(false);
                }
            }
            break;
        case TIMER_FULLSCREENCONTROLBARHIDER: {
            CPoint p;
            GetCursorPos(&p);

            CRect r;
            GetWindowRect(r);
            bool fCursorOutside = !r.PtInRect(p);

            CWnd* pWnd = WindowFromPoint(p);
            if (pWnd && (m_wndView == *pWnd || m_wndView.IsChild(pWnd) || fCursorOutside)) {
                if (AfxGetAppSettings().nShowBarsWhenFullScreenTimeOut >= 0) {
                    ShowControls(CS_NONE);
                }
            }
        }
        break;
        case TIMER_FULLSCREENMOUSEHIDER: {
            CPoint p;
            GetCursorPos(&p);

            CRect r;
            GetWindowRect(r);
            bool fCursorOutside = !r.PtInRect(p);
            CWnd* pWnd = WindowFromPoint(p);
            if (IsD3DFullScreenMode()) {
                if (pWnd && !m_bInOptions && *pWnd == *m_pFullscreenWnd) {
                    m_pFullscreenWnd->ShowCursor(false);
                }
                KillTimer(TIMER_FULLSCREENMOUSEHIDER);
            } else {
                if (pWnd && !m_bInOptions && (m_wndView == *pWnd || m_wndView.IsChild(pWnd) || fCursorOutside)) {
                    m_fHideCursor = true;
                    SetCursor(nullptr);
                }
            }
        }
        break;
        //统计量
        case TIMER_STATS: {
            //接收端质量信息：抖动，抖动，视音频同步情况等。。。
            if (m_pQP) {
                CString rate;
                rate.Format(_T("%.2f"), m_dSpeedRate);
                rate = _T("(") + rate + _T("x)");
                //信息
                CString info;
                int val = 0;
                //平均帧率
                m_pQP->get_AvgFrameRate(&val); // We hang here due to a lock that never gets released.
                info.Format(_T("%d.%02d %s"), val / 100, val % 100, rate);
                m_wndStatsBar.SetLine(ResStr(IDS_AG_FRAMERATE), info);

                int avg, dev;
                //抖动
                m_pQP->get_AvgSyncOffset(&avg);
                m_pQP->get_DevSyncOffset(&dev);
                info.Format(ResStr(IDS_STATSBAR_SYNC_OFFSET_FORMAT), avg, dev);
                m_wndStatsBar.SetLine(ResStr(IDS_STATSBAR_SYNC_OFFSET), info);
                //掉帧
                int drawn, dropped;
                m_pQP->get_FramesDrawn(&drawn);
                m_pQP->get_FramesDroppedInRenderer(&dropped);
                info.Format(IDS_MAINFRM_6, drawn, dropped);
                m_wndStatsBar.SetLine(ResStr(IDS_AG_FRAMES), info);
                //抖动
                m_pQP->get_Jitter(&val);
                info.Format(_T("%d ms"), val);
                m_wndStatsBar.SetLine(ResStr(IDS_STATSBAR_JITTER), info);
```

```
            }
            //缓存信息
            if (m_pBI) {
                CAtlList<CString> sl;
                //获取数量
                for (int i = 0, j = m_pBI->GetCount(); i < j; i++) {
                    int samples, size;
                    //获取缓存状态
                    if (S_OK == m_pBI->GetStatus(i, samples, size)) {
                        CString str;
                        str.Format(_T("[%d]: %03d/%d KB"), i, samples, size / 1024);
                        sl.AddTail(str);
                    }
                }

                if (!sl.IsEmpty()) {
                    CString str;
                    str.Format(_T("%s (p%u)"), Implode(sl, ' '), m_pBI->GetPriority());

                    m_wndStatsBar.SetLine(ResStr(IDS_AG_BUFFERS), str);
                }
            }
            //比特率信息
            CInterfaceList<IBitRateInfo> pBRIs;

            BeginEnumFilters(m_pGB, pEF, pBF) {
                BeginEnumPins(pBF, pEP, pPin) {
                    if (CComQIPtr<IBitRateInfo> pBRI = pPin) {
                        pBRIs.AddTail(pBRI);
                    }
                }
                EndEnumPins;

                if (!pBRIs.IsEmpty()) {
                    CAtlList<CString> sl;

                    POSITION pos = pBRIs.GetHeadPosition();
                    for (int i = 0; pos; i++) {
                        //比特率接口
                        IBitRateInfo* pBRI = pBRIs.GetNext(pos);
                        //当前比特率
                        DWORD cur = pBRI->GetCurrentBitRate() / 1000;
                        //平均比特率
                        DWORD avg = pBRI->GetAverageBitRate() / 1000;

                        if (avg == 0) {
                            continue;
                        }
                        //添加到字符串
                        CString str;
                        if (cur != avg) {
                            str.Format(_T("[%d]: %u/%u Kb/s"), i, avg, cur);
                        } else {
                            str.Format(_T("[%d]: %u Kb/s"), i, avg);
                        }
                        //加入
                        sl.AddTail(str);
                    }

                    if (!sl.IsEmpty()) {
                        m_wndStatsBar.SetLine(ResStr(IDS_STATSBAR_BITRATE), Implode(sl, ' ') + ResStr(IDS_STATSBAR_BITRATE_AVG_CUR));

                    }

                    break;
                }
            }
            EndEnumFilters;

            if (GetPlaybackMode() == PM_DVD) { // we also use this timer to update the info panel for DVD playback
                ULONG ulAvailable, ulCurrent;

                // Location

                CString Location('-');

                DVD_PLAYBACK_LOCATION2 loc;
                ULONG ulNumOfVolumes, ulVolume;
                DVD_DISC_SIDE Side;
                ULONG ulNumOfTitles;
                ULONG ulNumOfChapters;

                if (SUCCEEDED(m_pDVDI->GetCurrentLocation(&loc))
                        && SUCCEEDED(m_pDVDI->GetNumberOfChapters(loc.TitleNum, &ulNumOfChapters))
                        && SUCCEEDED(m_pDVDI->GetDVDVolumeInfo(&ulNumOfVolumes, &ulVolume, &Side, &ulNumOfTitles))) {
                    Location.Format(IDS_MAINFRM_9,
                                    ulVolume, ulNumOfVolumes,
                                    loc.TitleNum, ulNumOfTitles,
                                    loc.ChapterNum, ulNumOfChapters);
                    ULONG tsec = (loc.TimeCode.bHours * 3600)
                                 + (loc.TimeCode.bMinutes * 60)
```

```
                                + (loc.TimeCode.bSeconds);
                        /* This might not always work, such as on resume */
                        if (loc.ChapterNum != m_lCurrentChapter) {
                            m_lCurrentChapter = loc.ChapterNum;
                            m_lChapterStartTime = tsec;
                        } else {
                            /* If a resume point was used, and the user chapter jumps,
                            then it might do some funky time jumping.  Try to 'fix' the
                            chapter start time if this happens */
                            if (m_lChapterStartTime > tsec) {
                                m_lChapterStartTime = tsec;
                            }
                        }
                    }

                    m_wndInfoBar.SetLine(ResStr(IDS_INFOBAR_LOCATION), Location);

                    // Video

                    CString Video('-');

                    DVD_VideoAttributes VATR;

                    if (SUCCEEDED(m_pDVDI->GetCurrentAngle(&ulAvailable, &ulCurrent))
                            && SUCCEEDED(m_pDVDI->GetCurrentVideoAttributes(&VATR))) {
                        Video.Format(IDS_MAINFRM_10,
                                     ulCurrent, ulAvailable,
                                     VATR.ulSourceResolutionX, VATR.ulSourceResolutionY, VATR.ulFrameRate,
                                     VATR.ulAspectX, VATR.ulAspectY);
                    }

                    m_wndInfoBar.SetLine(ResStr(IDS_INFOBAR_VIDEO), Video);

                    // Audio

                    CString Audio('-');

                    DVD_AudioAttributes AATR;

                    if (SUCCEEDED(m_pDVDI->GetCurrentAudio(&ulAvailable, &ulCurrent))
                            && SUCCEEDED(m_pDVDI->GetAudioAttributes(ulCurrent, &AATR))) {
                        CString lang;
                        if (AATR.Language) {
                            int len = GetLocaleInfo(AATR.Language, LOCALE_SENGLANGUAGE, lang.GetBuffer(64), 64);
                            lang.ReleaseBufferSetLength(max(len - 1, 0));
                        } else {
                            lang.Format(IDS_AG_UNKNOWN, ulCurrent + 1);
                        }

                        switch (AATR.LanguageExtension) {
                            case DVD_AUD_EXT_NotSpecified:
                            default:
                                break;
                            case DVD_AUD_EXT_Captions:
                                lang += _T(" (Captions)");
                                break;
                            case DVD_AUD_EXT_VisuallyImpaired:
                                lang += _T(" (Visually Impaired)");
                                break;
                            case DVD_AUD_EXT_DirectorComments1:
                                lang += _T(" (Director Comments 1)");
                                break;
                            case DVD_AUD_EXT_DirectorComments2:
                                lang += _T(" (Director Comments 2)");
                                break;
                        }

                        CString format = GetDVDAudioFormatName(AATR);

                        Audio.Format(IDS_MAINFRM_11,
                                     lang,
                                     format,
                                     AATR.dwFrequency,
                                     AATR.bQuantization,
                                     AATR.bNumberOfChannels,
                                     (AATR.bNumberOfChannels > 1 ? ResStr(IDS_MAINFRM_13) : ResStr(IDS_MAINFRM_12)));

                        m_wndStatusBar.SetStatusBitmap(
                            AATR.bNumberOfChannels == 1 ? IDB_AUDIOTYPE_MONO
                            : AATR.bNumberOfChannels >= 2 ? IDB_AUDIOTYPE_STEREO
                            : IDB_AUDIOTYPE_NOAUDIO);
                    }

                    m_wndInfoBar.SetLine(ResStr(IDS_INFOBAR_AUDIO), Audio);

                    // Subtitles

                    CString Subtitles('-');

                    BOOL bIsDisabled;
                    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.     }
```


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