

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

2013年10月31日 00:33:25 阅读数：5693

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：MedialInfo选项卡（CPPageFileMedialInfo）](#)

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



前几篇文章分析了Media Player Classic - HC（mpc-hc）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）](#)

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

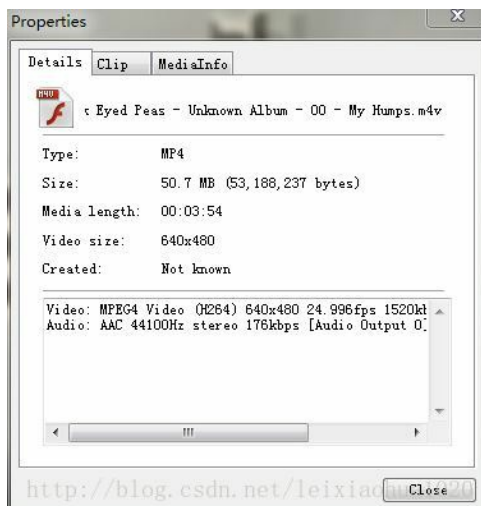
核心类分析完之后，分析了其他几个重要的类：

[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 5：MedialInfo选项卡（CPPageFileMedialInfo）](#)

本文分析一下mpc-hc的详细信息选项卡。在播放视频的时候，右键点击视频->选择“属性”后默认打开的就是该选项卡。一般情况下，该选项卡给出了正在播放的视频文件的一些基本参数：视频大小，分辨率，时长等。注意：详细信息选项卡和MedialInfo选项卡获得视频参数的原理是不一样的。详细信息选项卡是通过调用DirectShow函数接口而获得的视频的参数。而MedialInfo选项卡则是通过调用MedialInfo类库而获得视频的参数。

先看看选项卡长什么样子：



先来看看详细信息选项卡类的定义是什么样的吧。该类的定义位于PPageFileInfoDetails.h文件中。

```
[cpp]
1.  /* 雷霄骅
2.   * 中国传媒大学/数字电视技术
3.   * leixiaohua1020@126.com
4.   *
5.   */
6.  /*
7.   * (C) 2003-2006 Gabest
8.   * (C) 2006-2012 see Authors.txt
9.   *
10.  * This file is part of MPC-HC.
11.  *
12.  * MPC-HC is free software; you can redistribute it and/or modify
13.  * it under the terms of the GNU General Public License as published by
14.  * the Free Software Foundation; either version 3 of the License, or
15.  * (at your option) any later version.
16.  *
17.  * MPC-HC is distributed in the hope that it will be useful,
18.  * but WITHOUT ANY WARRANTY; without even the implied warranty of
19.  * MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the
20.  * GNU General Public License for more details.
21.  *
22.  * You should have received a copy of the GNU General Public License
23.  * along with this program. If not, see <http://www.gnu.org/licenses/>.
24.  *
25.  */
26.
27. #pragma once
28.
29. #include "../SubPic/ISubPic.h"
30. #include <afxwin.h>
31.
32. //取得一些基本信息
33. // CPPageFileInfoDetails dialog
34.
35. class CPPageFileInfoDetails : public CPropertyPage
36. {
37.     DECLARE_DYNAMIC(CPPageFileInfoDetails)
38.
39. private:
40.     CComPtr<IFilterGraph> m_pFG;
41.     CComPtr<ISubPicAllocatorPresenter> m_pCAP;
42.
43.     HICON m_hIcon;
44.
45.     void InitEncoding();
46.
47. public:
48.     CPPageFileInfoDetails(CString fn, IFilterGraph* pFG, ISubPicAllocatorPresenter* pCAP);
49.     virtual ~CPPageFileInfoDetails();
50.
51.     // Dialog Data
52.     enum { IDD = IDD_FILEPROPDDETAILS };
53.
54.     CStatic m_icon;
55.     CString m_fn;
56.     CString m_type;
57.     CString m_size;
58.     CString m_time;
59.     CString m_res;
60.     CString m_created;
61.     CEdit m_encoding;
62.
63. protected:
64.     virtual void DoDataExchange(CDataExchange* pDX);    // DDX/DDV support
65.     virtual BOOL OnInitDialog();
66.     virtual BOOL OnSetActive();
67.     virtual LRESULT OnSetPageFocus(WPARAM wParam, LPARAM lParam);
68.
69.     DECLARE_MESSAGE_MAP()
70.
71. public:
72. };
```

该类的定义和普通的MFC对话框类差不多，有以下几点是需要注意的：

1.以下两个变量是用于获得对话框上显示的相关的信息的：

```
CComPtr<IFilterGraph> m_pFG;
```

```
CComPtr<ISubPicAllocatorPresenter> m_pCAP;
```

2.有一系列的字符串变量：m_fn， m_type， m_size， m_time等用于存储获得的信息

详细信息 选项卡类的实现位于 PPageFileInfoDetails.cpp 文件中。

[cpp]  

```
1.  /* 雷霄骅
2.   * 中国传媒大学/数字电视技术
3.   * leixiaohua1020@126.com
4.   *
5.   */
6.  /*
7.   * (C) 2003-2006 Gabest
8.   * (C) 2006-2013 see Authors.txt
9.   *
10.  * This file is part of MPC-HC.
11.  *
12.  * MPC-HC is free software; you can redistribute it and/or modify
13.  * it under the terms of the GNU General Public License as published by
14.  * the Free Software Foundation; either version 3 of the License, or
15.  * (at your option) any later version.
16.  *
17.  * MPC-HC is distributed in the hope that it will be useful,
18.  * but WITHOUT ANY WARRANTY; without even the implied warranty of
19.  * MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the
20.  * GNU General Public License for more details.
21.  *
22.  * You should have received a copy of the GNU General Public License
23.  * along with this program. If not, see <http://www.gnu.org/licenses/>.
24.  *
25.  */
26.  //取得一些基本信息
27.  #include "stdafx.h"
28.  #include "mplayerc.h"
29.  #include "PPageFileInfoDetails.h"
30.  #include <atlbase.h>
31.  #include "DSUtil.h"
32.  #include "text.h"
33.  #include <d3d9.h>
34.  #include <vmr9.h>
35.  #include "moreuuids.h"
36.
37.
38.  // CPPageFileInfoDetails dialog
39.
40.  IMPLEMENT_DYNAMIC(CPPageFileInfoDetails, CPropertyPage)
41.  CPPageFileInfoDetails::CPPageFileInfoDetails(CString fn, IFilterGraph* pFG, ISubPicAllocatorPresenter* pCAP)
42.  : CPropertyPage(CPPageFileInfoDetails::IDD, CPPageFileInfoDetails::IDD)
43.  , m_fn(fn)
44.  , m_pFG(pFG)
45.  , m_pCAP(pCAP)
46.  , m_hIcon(nullptr)
47.  , m_type(ResStr(IDS_AG_NOT_KNOWN))
48.  , m_size(ResStr(IDS_AG_NOT_KNOWN))
49.  , m_time(ResStr(IDS_AG_NOT_KNOWN))
50.  , m_res(ResStr(IDS_AG_NOT_KNOWN))
51.  , m_created(ResStr(IDS_AG_NOT_KNOWN))
52.  {
53.  }
54.
55.  CPPageFileInfoDetails::~CPPageFileInfoDetails()
56.  {
57.  if (m_hIcon) {
58.  DestroyIcon(m_hIcon);
59.  }
60.  }
61.
62.  void CPPageFileInfoDetails::DoDataExchange(CDataExchange* pDX)
63.  {
64.  __super::DoDataExchange(pDX);
65.  DDX_Control(pDX, IDC_DEFAULTICON, m_icon);
66.  DDX_Text(pDX, IDC_EDIT1, m_fn);
67.  DDX_Text(pDX, IDC_EDIT4, m_type);
68.  DDX_Text(pDX, IDC_EDIT3, m_size);
69.  DDX_Text(pDX, IDC_EDIT2, m_time);
70.  DDX_Text(pDX, IDC_EDIT5, m_res);
71.  DDX_Text(pDX, IDC_EDIT6, m_created);
72.  DDX_Control(pDX, IDC_EDIT7, m_encoding);
73.  }
74.
75.  #define SETPAGEFOCUS (WM_APP + 252) // arbitrary number, can be changed if necessary
76.
77.  BEGIN_MESSAGE_MAP(CPPageFileInfoDetails, CPropertyPage)
78.  ON_MESSAGE(SETPAGEFOCUS, OnSetPageFocus)
79.  END_MESSAGE_MAP()
80.
81.  // CPPageFileInfoDetails message handlers
82.
83.  static bool GetProperty(IFilterGraph* pFG, LPCOLESTR propName, VARIANT* vt)
84.  {
85.  BeginEnumFilters(pFG, pEF, pBF) {
86.  if (CComQIPtr<IPropertyBag> pPB = pBF)
87.  if (SUCCEEDED(pPB->Read(propName, vt, nullptr))) {
88.  return true;
89.  }
90.  }
```

```

90.         }
91.         EndEnumFilters;
92.
93.         return false;
94.     }
95.
96.     static CString FormatDateTime(FILETIME tm)
97.     {
98.         SYSTEMTIME st;
99.         FileTimeToSystemTime(&tm, &st);
100.        TCHAR buff[256];
101.        GetDateFormat(LOCALE_USER_DEFAULT, DATE_LONGDATE, &st, nullptr, buff, _countof(buff));
102.        CString ret(buff);
103.        ret += _T(" ");
104.        GetTimeFormat(LOCALE_USER_DEFAULT, 0, &st, nullptr, buff, _countof(buff));
105.        ret += buff;
106.        return ret;
107.    }
108.
109.    BOOL CPPageFileInfoDetails::OnInitDialog()
110.    {
111.        __super::OnInitDialog();
112.
113.        if (m_fn.IsEmpty()) {
114.            BeginEnumFilters(m_pFG, pEF, pBF) {
115.                CComQIPtr<IFileSourceFilter> pFSF = pBF;
116.                if (pFSF) {
117.                    LPOLESTR pFN = nullptr;
118.                    AM_MEDIA_TYPE mt;
119.                    if (SUCCEEDED(pFSF->GetCurFile(&pFN, &mt)) && pFN && *pFN) {
120.                        m_fn = CStringW(pFN);
121.                        CoTaskMemFree(pFN);
122.                    }
123.                    break;
124.                }
125.            }
126.            EndEnumFilters;
127.        }
128.
129.        CString ext = m_fn.Left(m_fn.Find(_T(":/")) + 1).TrimRight(':');
130.        if (ext.IsEmpty() || !ext.CompareNoCase(_T("file"))) {
131.            ext = _T(".") + m_fn.Mid(m_fn.ReverseFind('.') + 1);
132.        }
133.
134.        m_hIcon = LoadIcon(m_fn, false);
135.        if (m_hIcon) {
136.            m_icon.SetIcon(m_hIcon);
137.        }
138.
139.        if (!LoadType(ext, m_type)) {
140.            m_type.LoadString(IDS_AG_NOT_KNOWN);
141.        }
142.
143.        CComVariant vt;
144.        if (::GetProperty(m_pFG, L"CurFile.TimeCreated", &vt)) {
145.            if (V_VT(&vt) == VT_UI8) {
146.                ULARGE_INTEGER uli;
147.                uli.QuadPart = V_UI8(&vt);
148.
149.                FILETIME ft;
150.                ft.dwLowDateTime = uli.LowPart;
151.                ft.dwHighDateTime = uli.HighPart;
152.
153.                m_created = FormatDateTime(ft);
154.            }
155.        }
156.
157.        WIN32_FIND_DATA wfd;
158.        HANDLE hFind = FindFirstFile(m_fn, &wfd);
159.        if (hFind != INVALID_HANDLE_VALUE) {
160.            FindClose(hFind);
161.
162.            __int64 size = (__int64(wfd.nFileSizeHigh) << 32) | wfd.nFileSizeLow;
163.            const int MAX_FILE_SIZE_BUFFER = 65;
164.            WCHAR szFileSize[MAX_FILE_SIZE_BUFFER];
165.            StrFormatByteSizeW(size, szFileSize, MAX_FILE_SIZE_BUFFER);
166.            CString szByteSize;
167.            szByteSize.Format(_T("%I64d"), size);
168.            m_size.Format(_T("%s (%s bytes)"), szFileSize, FormatNumber(szByteSize));
169.
170.            if (m_created.IsEmpty()) {
171.                m_created = FormatDateTime(wfd.ftCreationTime);
172.            }
173.        }
174.        //获取时常
175.        REFERENCE_TIME rtDur = 0;
176.        CComQIPtr<IMediaSeeking> pMS = m_pFG;
177.        if (pMS && SUCCEEDED(pMS->GetDuration(&rtDur)) && rtDur > 0) {
178.            m_time = ReftimeToString2(rtDur);
179.        }
180.
181.        CSize wh(0, 0). arxv(0, 0):

```

```

182. //获取宽和高
183. if (m_pCAP) {
184.     wh = m_pCAP->GetVideoSize(false);
185.     arxy = m_pCAP->GetVideoSize(true);
186. } else {
187.     if (CComQIPtr<IBasicVideo> pBV = m_pFG) {
188.         if (SUCCEEDED(pBV->GetVideoSize(&wh.cx, &wh.cy))) {
189.             if (CComQIPtr<IBasicVideo2> pBV2 = m_pFG) {
190.                 pBV2->GetPreferredAspectRatio(&arxy.cx, &arxy.cy);
191.             }
192.         } else {
193.             wh.SetSize(0, 0);
194.         }
195.     }
196.
197.     if (wh.cx == 0 && wh.cy == 0) {
198.         BeginEnumFilters(m_pFG, pEF, pBF) {
199.             if (CComQIPtr<IBasicVideo> pBV = pBF) {
200.                 pBV->GetVideoSize(&wh.cx, &wh.cy);
201.                 if (CComQIPtr<IBasicVideo2> pBV2 = pBF) {
202.                     pBV2->GetPreferredAspectRatio(&arxy.cx, &arxy.cy);
203.                 }
204.                 break;
205.             } else if (CComQIPtr<IVMRWindowlessControl> pWC = pBF) {
206.                 pWC->GetNativeVideoSize(&wh.cx, &wh.cy, &arxy.cx, &arxy.cy);
207.                 break;
208.             } else if (CComQIPtr<IVMRWindowlessControl9> pWC9 = pBF) {
209.                 pWC9->GetNativeVideoSize(&wh.cx, &wh.cy, &arxy.cx, &arxy.cy);
210.                 break;
211.             }
212.         }
213.         EndEnumFilters;
214.     }
215. }
216.
217. if (wh.cx > 0 && wh.cy > 0) {
218.     m_res.Format(_T("%dx%d"), wh.cx, wh.cy);
219.
220.     int gcd = GCD(arxy.cx, arxy.cy);
221.     if (gcd > 1) {
222.         arxy.cx /= gcd;
223.         arxy.cy /= gcd;
224.     }
225.
226.     if (arxy.cx > 0 && arxy.cy > 0 && arxy.cx * wh.cy != arxy.cy * wh.cx) {
227.         CString ar;
228.         ar.Format(_T(" (AR %d:%d)", arxy.cx, arxy.cy);
229.         m_res += ar;
230.     }
231. }
232.
233. m_fn.TrimRight('/');
234. m_fn.Replace('\\', '/');
235. m_fn = m_fn.Mid(m_fn.ReverseFind('/') + 1);
236.
237. UpdateData(FALSE);
238.
239. InitEncoding();
240.
241. m_pFG = nullptr;
242. m_pCAP = nullptr;
243.
244. return TRUE; // return TRUE unless you set the focus to a control
245. // EXCEPTION: OCX Property Pages should return FALSE
246. }
247.
248. BOOL CPageFileInfoDetails::OnSetActive()
249. {
250.     BOOL ret = __super::OnSetActive();
251.     PostMessage(SETPAGEFOCUS, 0, 0L);
252.     return ret;
253. }
254.
255. LRESULT CPageFileInfoDetails::OnSetPageFocus(WPARAM wParam, LPARAM lParam)
256. {
257.     CPropertySheet* psheet = (CPropertySheet*) GetParent();
258.     psheet->GetTabControl()->SetFocus();
259.     return 0;
260. }
261.
262. void CPageFileInfoDetails::InitEncoding()
263. {
264.     CAtlList<CString> sl;
265.
266.     BeginEnumFilters(m_pFG, pEF, pBF) {
267.         CComPtr<IBaseFilter> pUSBF = GetUpStreamFilter(pBF);
268.
269.         if (GetCLSID(pBF) == CLSID_NetShowSource) {
270.             continue;
271.         } else if (GetCLSID(pUSBF) != CLSID_NetShowSource) {
272.             if (IPin* pPin = GetFirstPin(pBF, PINDIR_INPUT)) {

```

```

273.         CMediaType mt;
274.         if (FAILED(pPin->ConnectionMediaType(&mt)) || mt.majortype != MEDIATYPE_Stream) {
275.             continue;
276.         }
277.     }
278.
279.     if (IPin* pPin = GetFirstPin(pBF, PINDIR_OUTPUT)) {
280.         CMediaType mt;
281.         if (SUCCEEDED(pPin->ConnectionMediaType(&mt)) && mt.majortype == MEDIATYPE_Stream) {
282.             continue;
283.         }
284.     }
285. }
286.
287. BeginEnumPins(pBF, pEP, pPin) {
288.     CMediaTypeEx mt;
289.     PIN_DIRECTION dir;
290.     if (FAILED(pPin->QueryDirection(&dir)) || dir != PINDIR_OUTPUT
291.         || FAILED(pPin->ConnectionMediaType(&mt))) {
292.         continue;
293.     }
294.
295.     CString str = mt.ToString();
296.
297.     if (!str.IsEmpty()) {
298.         if (mt.majortype == MEDIATYPE_Video) { // Sort streams, set Video streams at head
299.             bool found_video = false;
300.             for (POSITION pos = sl.GetTailPosition(); pos; sl.GetPrev(pos)) {
301.                 CString Item = sl.GetAt(pos);
302.                 if (!Item.Find(_T("Video:"))) {
303.                     sl.InsertAfter(pos, str + CString(L" [" + GetPinName(pPin) + L"]"));
304.                     found_video = true;
305.                     break;
306.                 }
307.             }
308.             if (!found_video) {
309.                 sl.AddHead(str + CString(L" [" + GetPinName(pPin) + L"]"));
310.             }
311.         } else {
312.             sl.AddTail(str + CString(L" [" + GetPinName(pPin) + L"]"));
313.         }
314.     }
315. }
316. EndEnumPins;
317. }
318. EndEnumFilters;
319.
320. CString text = Implode(sl, '\n');
321. text.Replace(_T("\n"), _T("\r\n"));
322. m_encoding.SetWindowText(text);
323. }

```

从代码中可以看出，CPPageFileInfoDetails的功能是在OnInitDialog()中完成的。通过调用DirectShow相应的接口，分别获取了视频的宽高，时长等信息。

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/13297621>

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

个人分类： [MPC-HC](#)

所属专栏： [开源多媒体项目源代码分析](#)

