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：MediaInfo选项卡 （CPPageFileMediaInfo）

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）

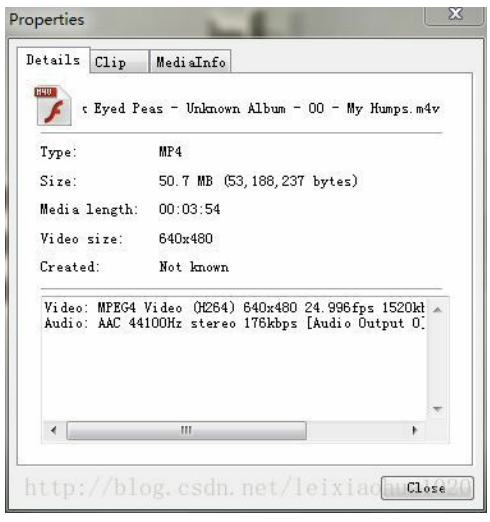
核心类分析完之后，分析了其他几个重要的类：

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：MediaInfo选项卡 （CPPageFileMediaInfo）

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MediaInfo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MediaInfo选项卡则是通过调用MediaInfo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
/* 雷霄骅
 * 中国传媒大学/数字电视技术
 * leixiaohua1020@126.com
 *
 */
/*
 * (C) 2003-2006 Gabest
 * (C) 2006-2012 see Authors.txt
 *
 * This file is part of MPC-HC.
 *
 * MPC-HC is free software; you can redistribute it and/or modify
 * it under the terms of the GNU General Public License as published by
 * the Free Software Foundation; either version 3 of the License, or
 * (at your option) any later version.
 *
 * MPC-HC is distributed in the hope that it will be useful,
 * but WITHOUT ANY WARRANTY; without even the implied warranty of
 * MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.  See the
 * GNU General Public License for more details.
 *
 * You should have received a copy of the GNU General Public License
 * along with this program.  If not, see <http://www.gnu.org/licenses/>.
 *
 */

#pragma once

#include "../SubPic/ISubPic.h"
#include <afxwin.h>

//取得一些基本信息
// CPPageFileInfoDetails dialog

class CPPageFileInfoDetails : public CPropertyPage
{
    DECLARE_DYNAMIC(CPPageFileInfoDetails)

private:
    CComPtr<IFilterGraph> m_pFG;
    CComPtr<ISubPicAllocatorPresenter> m_pCAP;

    HICON m_hIcon;

    void InitEncoding();

public:
    CPPageFileInfoDetails(CString fn, IFilterGraph* pFG, ISubPicAllocatorPresenter* pCAP);
    virtual ~CPPageFileInfoDetails();

    // Dialog Data
    enum { IDD = IDD_FILEPROPDETAILS };

    CStatic m_icon;
    CString m_fn;
    CString m_type;
    CString m_size;
    CString m_time;
    CString m_res;
    CString m_created;
    CEdit   m_encoding;

protected:
    virtual void DoDataExchange(CDataExchange* pDX);    // DDX/DDV support
    virtual BOOL OnInitDialog();
    virtual BOOL OnSetActive();
    virtual LRESULT OnSetPageFocus(WPARAM wParam, LPARAM lParam);

    DECLARE_MESSAGE_MAP()

public:
};
```

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
/* 雷霄骅
 * 中国传媒大学/数字电视技术
 * leixiaohua1020@126.com
 *
 */
/*
 * (C) 2003-2006 Gabest
 * (C) 2006-2013 see Authors.txt
 *
 * This file is part of MPC-HC.
 *
 * MPC-HC is free software; you can redistribute it and/or modify
 * it under the terms of the GNU General Public License as published by
 * the Free Software Foundation; either version 3 of the License, or
 * (at your option) any later version.
 *
 * MPC-HC is distributed in the hope that it will be useful,
 * but WITHOUT ANY WARRANTY; without even the implied warranty of
 * MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.  See the
 * GNU General Public License for more details.
 *
 * You should have received a copy of the GNU General Public License
 * along with this program.  If not, see <http://www.gnu.org/licenses/>.
 *
 */
//取得一些基本信息
#include "stdafx.h"
#include "mplayerc.h"
#include "PPageFileInfoDetails.h"
#include <atlbase.h>
#include "DSUtil.h"
#include "text.h"
#include <d3d9.h>
#include <vmr9.h>
#include "moreuuids.h"


// CPPageFileInfoDetails dialog

IMPLEMENT_DYNAMIC(CPPageFileInfoDetails, CPropertyPage)
CPPageFileInfoDetails::CPPageFileInfoDetails(CString fn, IFilterGraph* pFG, ISubPicAllocatorPresenter* pCAP)
    : CPropertyPage(CPPageFileInfoDetails::IDD, CPPageFileInfoDetails::IDD)
    , m_fn(fn)
    , m_pFG(pFG)
    , m_pCAP(pCAP)
    , m_hIcon(nullptr)
    , m_type(ResStr(IDS_AG_NOT_KNOWN))
    , m_size(ResStr(IDS_AG_NOT_KNOWN))
    , m_time(ResStr(IDS_AG_NOT_KNOWN))
    , m_res(ResStr(IDS_AG_NOT_KNOWN))
    , m_created(ResStr(IDS_AG_NOT_KNOWN))
{
}

CPPageFileInfoDetails::~CPPageFileInfoDetails()
{
    if (m_hIcon) {
        DestroyIcon(m_hIcon);
    }
}

void CPPageFileInfoDetails::DoDataExchange(CDataExchange* pDX)
{
    __super::DoDataExchange(pDX);
    DDX_Control(pDX, IDC_DEFAULTICON, m_icon);
    DDX_Text(pDX, IDC_EDIT1, m_fn);
    DDX_Text(pDX, IDC_EDIT4, m_type);
    DDX_Text(pDX, IDC_EDIT3, m_size);
    DDX_Text(pDX, IDC_EDIT2, m_time);
    DDX_Text(pDX, IDC_EDIT5, m_res);
    DDX_Text(pDX, IDC_EDIT6, m_created);
    DDX_Control(pDX, IDC_EDIT7, m_encoding);
}

#define SETPAGEFOCUS (WM_APP + 252) // arbitrary number, can be changed if necessary

BEGIN_MESSAGE_MAP(CPPageFileInfoDetails, CPropertyPage)
    ON_MESSAGE(SETPAGEFOCUS, OnSetPageFocus)
END_MESSAGE_MAP()

// CPPageFileInfoDetails message handlers

static bool GetProperty(IFilterGraph* pFG, LPCOLESTR propName, VARIANT* vt)
{
    BeginEnumFilters(pFG, pEF, pBF) {
        if (CComQIPtr<IPropertyBag> pPB = pBF)
            if (SUCCEEDED(pPB->Read(propName, vt, nullptr))) {
                return true;
            }
    }
```

```
    }
    EndEnumFilters;

    return false;
}

static CString FormatDateTime(FILETIME tm)
{
    SYSTEMTIME st;
    FileTimeToSystemTime(&tm, &st);
    TCHAR buff[256];
    GetDateFormat(LOCALE_USER_DEFAULT, DATE_LONGDATE, &st, nullptr, buff, _countof(buff));
    CString ret(buff);
    ret += _T(" ");
    GetTimeFormat(LOCALE_USER_DEFAULT, 0, &st, nullptr, buff, _countof(buff));
    ret += buff;
    return ret;
}

BOOL CPPageFileInfoDetails::OnInitDialog()
{
    __super::OnInitDialog();

    if (m_fn.IsEmpty()) {
        BeginEnumFilters(m_pFG, pEF, pBF) {
            CComQIPtr<IFileSourceFilter> pFSF = pBF;
            if (pFSF) {
                LPOLESTR pFN = nullptr;
                AM_MEDIA_TYPE mt;
                if (SUCCEEDED(pFSF->GetCurFile(&pFN, &mt)) && pFN && *pFN) {
                    m_fn = CStringW(pFN);
                    CoTaskMemFree(pFN);
                }
                break;
            }
        }
        EndEnumFilters;
    }

    CString ext = m_fn.Left(m_fn.Find(_T("://")) + 1).TrimRight(':');
    if (ext.IsEmpty() || !ext.CompareNoCase(_T("file"))) {
        ext = _T(".") + m_fn.Mid(m_fn.ReverseFind('.') + 1);
    }

    m_hIcon = LoadIcon(m_fn, false);
    if (m_hIcon) {
        m_icon.SetIcon(m_hIcon);
    }

    if (!LoadType(ext, m_type)) {
        m_type.LoadString(IDS_AG_NOT_KNOWN);
    }

    CComVariant vt;
    if (::GetProperty(m_pFG, L"CurFile.TimeCreated", &vt)) {
        if (V_VT(&vt) == VT_UI8) {
            ULARGE_INTEGER uli;
            uli.QuadPart = V_UI8(&vt);

            FILETIME ft;
            ft.dwLowDateTime = uli.LowPart;
            ft.dwHighDateTime = uli.HighPart;

            m_created = FormatDateTime(ft);
        }
    }

    WIN32_FIND_DATA wfd;
    HANDLE hFind = FindFirstFile(m_fn, &wfd);
    if (hFind != INVALID_HANDLE_VALUE) {
        FindClose(hFind);

        __int64 size = (__int64(wfd.nFileSizeHigh) << 32) | wfd.nFileSizeLow;
        const int MAX_FILE_SIZE_BUFFER = 65;
        WCHAR szFileSize[MAX_FILE_SIZE_BUFFER];
        StrFormatByteSizeW(size, szFileSize, MAX_FILE_SIZE_BUFFER);
        CString szByteSize;
        szByteSize.Format(_T("%I64d"), size);
        m_size.Format(_T("%s (%s bytes)"), szFileSize, FormatNumber(szByteSize));

        if (m_created.IsEmpty()) {
            m_created = FormatDateTime(wfd.ftCreationTime);
        }
    }
    //获取时常
    REFERENCE_TIME rtDur = 0;
    CComQIPtr<IMediaSeeking> pMS = m_pFG;
    if (pMS && SUCCEEDED(pMS->GetDuration(&rtDur)) && rtDur > 0) {
        m_time = ReftimeToString2(rtDur);
    }

    CSize wh(0, 0), arxy(0, 0);
```

```
    //获取宽和高
    if (m_pCAP) {
        wh = m_pCAP->GetVideoSize(false);
        arxy = m_pCAP->GetVideoSize(true);
    } else {
        if (CComQIPtr<IBasicVideo> pBV = m_pFG) {
            if (SUCCEEDED(pBV->GetVideoSize(&wh.cx, &wh.cy))) {
                if (CComQIPtr<IBasicVideo2> pBV2 = m_pFG) {
                    pBV2->GetPreferredAspectRatio(&arxy.cx, &arxy.cy);
                }
            } else {
                wh.SetSize(0, 0);
            }
        }

        if (wh.cx == 0 && wh.cy == 0) {
            BeginEnumFilters(m_pFG, pEF, pBF) {
                if (CComQIPtr<IBasicVideo> pBV = pBF) {
                    pBV->GetVideoSize(&wh.cx, &wh.cy);
                    if (CComQIPtr<IBasicVideo2> pBV2 = pBF) {
                        pBV2->GetPreferredAspectRatio(&arxy.cx, &arxy.cy);
                    }
                    break;
                } else if (CComQIPtr<IVMRWindowlessControl> pWC = pBF) {
                    pWC->GetNativeVideoSize(&wh.cx, &wh.cy, &arxy.cx, &arxy.cy);
                    break;
                } else if (CComQIPtr<IVMRWindowlessControl9> pWC9 = pBF) {
                    pWC9->GetNativeVideoSize(&wh.cx, &wh.cy, &arxy.cx, &arxy.cy);
                    break;
                }
            }
            EndEnumFilters;
        }
    }

    if (wh.cx > 0 && wh.cy > 0) {
        m_res.Format(_T("%dx%d"), wh.cx, wh.cy);

        int gcd = GCD(arxy.cx, arxy.cy);
        if (gcd > 1) {
            arxy.cx /= gcd;
            arxy.cy /= gcd;
        }

        if (arxy.cx > 0 && arxy.cy > 0 && arxy.cx * wh.cy != arxy.cy * wh.cx) {
            CString ar;
            ar.Format(_T(" (AR %d:%d)"), arxy.cx, arxy.cy);
            m_res += ar;
        }
    }

    m_fn.TrimRight('/');
    m_fn.Replace('\\', '/');
    m_fn = m_fn.Mid(m_fn.ReverseFind('/') + 1);

    UpdateData(FALSE);

    InitEncoding();

    m_pFG = nullptr;
    m_pCAP = nullptr;

    return TRUE;  // return TRUE unless you set the focus to a control
    // EXCEPTION: OCX Property Pages should return FALSE
}

BOOL CPPageFileInfoDetails::OnSetActive()
{
    BOOL ret = __super::OnSetActive();
    PostMessage(SETPAGEFOCUS, 0, 0L);
    return ret;
}

LRESULT CPPageFileInfoDetails::OnSetPageFocus(WPARAM wParam, LPARAM lParam)
{
    CPropertySheet* psheet = (CPropertySheet*) GetParent();
    psheet->GetTabControl()->SetFocus();
    return 0;
}

void CPPageFileInfoDetails::InitEncoding()
{
    CAtlList<CString> sl;

    BeginEnumFilters(m_pFG, pEF, pBF) {
        CComPtr<IBaseFilter> pUSBF = GetUpStreamFilter(pBF);

        if (GetCLSID(pBF) == CLSID_NetShowSource) {
            continue;
        } else if (GetCLSID(pUSBF) != CLSID_NetShowSource) {
            if (IPin* pPin = GetFirstPin(pBF, PINDIR_INPUT)) {
```

```
                CMediaType mt;
                if (FAILED(pPin->ConnectionMediaType(&mt)) || mt.majortype != MEDIATYPE_Stream) {
                    continue;
                }
            }

            if (IPin* pPin = GetFirstPin(pBF, PINDIR_OUTPUT)) {
                CMediaType mt;
                if (SUCCEEDED(pPin->ConnectionMediaType(&mt)) && mt.majortype == MEDIATYPE_Stream) {
                    continue;
                }
            }
        }

        BeginEnumPins(pBF, pEP, pPin) {
            CMediaTypeEx mt;
            PIN_DIRECTION dir;
            if (FAILED(pPin->QueryDirection(&dir)) || dir != PINDIR_OUTPUT
                    || FAILED(pPin->ConnectionMediaType(&mt))) {
                continue;
            }

            CString str = mt.ToString();

            if (!str.IsEmpty()) {
                if (mt.majortype == MEDIATYPE_Video) { // Sort streams, set Video streams at head
                    bool found_video = false;
                    for (POSITION pos = sl.GetTailPosition(); pos; sl.GetPrev(pos)) {
                        CString Item = sl.GetAt(pos);
                        if (!Item.Find(_T("Video:"))) {
                            sl.InsertAfter(pos, str + CString(L" [" + GetPinName(pPin) + L"]"));
                            found_video = true;
                            break;
                        }
                    }
                    if (!found_video) {
                        sl.AddHead(str + CString(L" [" + GetPinName(pPin) + L"]"));
                    }
                } else {
                    sl.AddTail(str + CString(L" [" + GetPinName(pPin) + L"]"));
                }
            }
        }
        EndEnumPins;
    }
    EndEnumFilters;

    CString text = Implode(sl, '\n');
    text.Replace(_T("\n"), _T("\r\n"));
    m_encoding.SetWindowText(text);
}
```

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。

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

我的邮箱:liushidc@163.com