Scrollable Dialog Demo

William, Jan, 16, 2022

摘要

Scrollable Dialog 是最常见的应用软件界面格式. 但在我手边的多本有关 MFC 编程的书中竟然没有一本提到该怎么制作. 本文介绍网上下载的一个比较好的例子P01. 该例作者参考了MFC 有关 Scrollable Dialog 的权威文本Q262954 制作了这个范例. 这个范例有三处值得参考:

1) 怎么制作 Scrollable Dialog
2) 在 Dialog 中再加子 Dialog.
3) 用 Mouse 移动 Dialog 中的内容.

I. ScrollableChildDialog Demo 简介

https://www.codeproject.com/Articles/4087/How-to-implement-a-scrollable-child-dialog

图1. Scrollable Child Dialog

The code is really simple and is strongly based on the MSDN article HOWTO: Create a Resizable Dialog Box with Scroll Bars (Q262954). I simply created the scrollable dialog template choosing the `WS_CHILD` and `WS_VSCROLL` styles and used class wizard to add `WM_SIZE`, `WM_MOUSEWHEEL` and `WM_VSCROLL` messages. To use the code in your project follow these steps:

1. Add *dlgscrollable.h* and *dlgscrollable.cpp* to your project
2. Create your own dialog resource template and replace the resource identifier in the header file

3. In the parent window, add a member variable like `CDlgScrollable* m_pdlgScrollable` and in the `OnInitDialog` function, create the new `CDlgScrollable` object.
4. In the parent dialog resource template, you can use a static as a placeholder to calc the rectangle of my scrollable child control.
5. You can easily change the "hand" cursors, loading different resources as needed

II. 该范例有三处值得参考

1. Scrollable Dialog class CDlgScrollable

范例作者参考了MFC Scrollable Dialog 的权威文本Q262954, 制作了这个 Scrollable Dialog class. 详细情况可以参考这个文本:

> Q262954: HOWTO: Create a Resizeable Dialog Box with Scroll Bars
> https://jeffpar.github.io/kbarchive/kb/262/Q262954/

但是作者在移动屏幕的 Function OnVScroll()中做了一些修改 使移动的效果比原文描述的还要平稳一些. 为了方便读者使用, 作者还把所有的改动做成一个 template class, CDlgScrollable. 读者使用时只要把该 class 两个文件 DlgScrollable.cpp 和 DlgScrollable.h 加到你的 MFC Project 中间就可以调用了. 在文本最后的 Appendix 中有这两个文件的 Copy.

这里要附加指出的是: 因为作者在 class CDlgScrollable 增加了一个用光标移动内容的功能, 所以在调用class CDlgScrollable 的时候要提供两个光标图案 ,IDC_CURSOR1 和 IDC_CURSOR2. 在第一次调用的时候可以避免这个麻烦, 也就是把程序中关于这两个光标的内容暂时注解掉. 详细可见本文 Page 3, “3. 用 Mouse 移动子 Dialog 中的内容.”

2. Dialog 中再加上子 Dialog.

从图1中可以看到这个例子实际上是一个 Scrollable Child Dialog. 这里描述一下如何在 Dialog 中间建立一个子 Dialog.

1) 首先在这里假定: 父 Dialog 是 CP01Dlg; 子 Dialog是 CDlgScrollable.
2) 在父 Dialog 中建立一个 Picture Control (IDC_PLACEHOLDER). 这个

control 被用做一个 container 来装载子 Dialog.

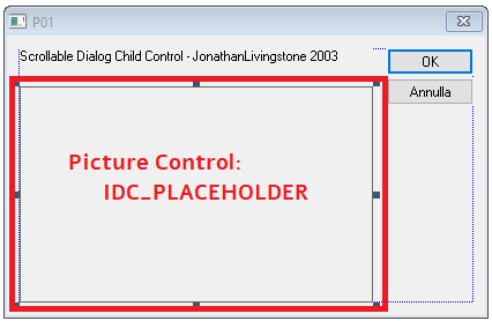

图2. 在父 Dialog 中建立一个 Picture Control (IDC_PLACEHOLDER).

3) 在父 Dialog H (CP01Dlg.h) 文件中加入

```
class CP01Dlg : public CDialog
{
// Construction
public:
 CP01Dlg(CWnd* pParent = NULL);     // standard constructor

 ~CP01Dlg()
 {
        delete m_pdlgScroll;
 }
 CDlgScrollable*  m_pdlgScroll;
```

4) 在父 Dialog CPP (CP01Dlg.cpp) 文件的 OnInitDialog() 中加入

```
BOOL CP01Dlg::OnInitDialog()
{
 CDialog::OnInitDialog();
 … …
 … …

 // TODO: Add extra initialization here

 // Create Child dialog class instance
 m_pdlgScroll = new CDlgScrollable(this);

 // Get Parent RC
 CRect rc;
 GetDlgItem(IDC_PLACEHOLDER)->GetWindowRect(rc);
 ScreenToClient(&rc);

 // Move Child window into farther clather.
 m_pdlgScroll->MoveWindow(rc);

 return TRUE;
}
```

3. 用 Mouse 移动子 Dialog 中的内容.

除了用 Scroll Bar 来移动内容以外，在这个程序中还加了一段用光标来移动内容的程序，见图3.

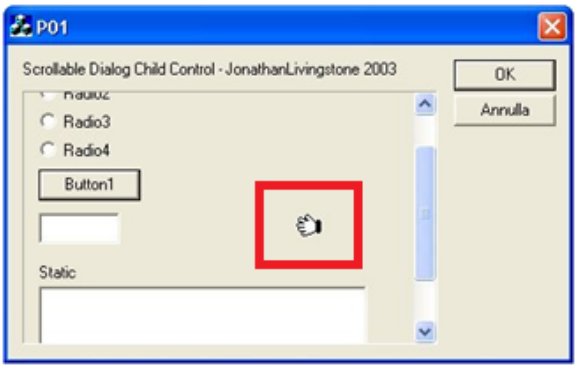

图3. 用光标来移动内容

为了用光标来移动内容，改程序动用了以下的 messages:

```
ON_WM_MOUSEWHEEL()
ON_WM_LBUTTONDOWN()
ON_WM_LBUTTONUP()
ON_WM_MOUSEMOVE()
ON_WM_KILLFOCUS()
ON_WM_SETCURSOR()
```

程序有如下改动:

1) 在 CDlgScrollable.h 加入:

```
class CDlgScrollable : public CDialog
{
// Construction
public:
      HCURSOR m_hCursor1, m_hCursor2;
      …….
```

2) 建立光标

A) 在 MFC Project Resource 中间建立两个 32x32 光标IDC_CURSOR1 和 IDC_CURSOR2.

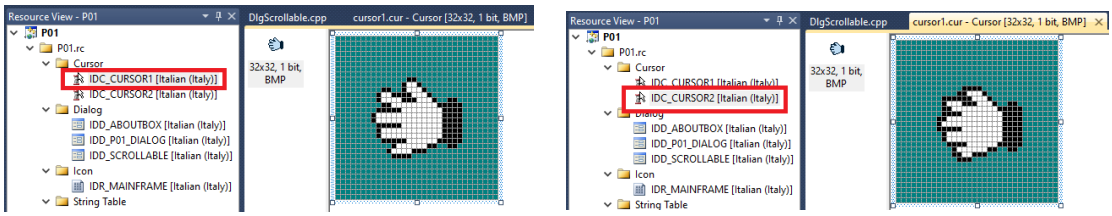

图4. 建立两个光标 IDC_CURSOR1 和 IDC_CURSOR2

B) 在 OnInitDialog()中加入 Load 两个光标:

```
BOOL CDlgScrollable::OnInitDialog()
{
  CDialog::OnInitDialog();

  m_hCursor1=AfxGetApp()->LoadCursor(IDC_CURSOR1);
  m_hCursor2=AfxGetApp()->LoadCursor(IDC_CURSOR2);
   … …
}
```

2) 用上述的 Messages 产生多个 Functions.
列如:
OnMouseWheel(), OnLButtonDown(), …, 等等. 所以就有下面这些 code.

请注意, 在 OnMouseMove() function 中用了与 OnVScroll() 同样的 SetScrollPos() 方式设置 Scroll Bar 位置和用 ScrollWindow() 这个功能移动内容. 因此在用光标移动内容的时候是与 OnVScroll() 同步的.

```
BOOL CDlgScrollable::OnMouseWheel(UINT nFlags, short zDelta, CPoint pt)
{
       SetScrollPos(SB_VERT,m_nScrollPos,TRUE);
       ScrollWindow(0,-nDelta);
  … …
}
```

```
BOOL CDlgScrollable::OnMouseWheel(UINT nFlags, short zDelta, CPoint pt)
{
       int nMaxPos = m_rcOriginalRect.Height() - m_nCurHeight;

       if (zDelta<0)
       {
              if (m_nScrollPos < nMaxPos)
              {
                     zDelta = min(max(nMaxPos/20,5),nMaxPos-m_nScrollPos);

                     m_nScrollPos += zDelta;
                     SetScrollPos(SB_VERT,m_nScrollPos,TRUE);
                     ScrollWindow(0,-zDelta);
              }
       }
       else
       {
              if (m_nScrollPos > 0)
              {
                     zDelta = -min(max(nMaxPos/20,5),m_nScrollPos);

                     m_nScrollPos += zDelta;
                     SetScrollPos(SB_VERT,m_nScrollPos,TRUE);
                     ScrollWindow(0,-zDelta);
              }
       }
```

```
	return CDialog::OnMouseWheel(nFlags, zDelta, pt);
}

void CDlgScrollable::OnLButtonDown(UINT nFlags, CPoint point)
{
	m_bDragging=TRUE;
	SetCapture();

	m_ptDragPoint=point;

	SetCursor(m_hCursor2);

	CDialog::OnLButtonDown(nFlags, point);
}

void CDlgScrollable::OnLButtonUp(UINT nFlags, CPoint point)
{
	EndDrag();

	CDialog::OnLButtonUp(nFlags, point);
}

void CDlgScrollable::OnMouseMove(UINT nFlags, CPoint point)
{
	if (m_bDragging)
	{
		int nDelta=m_ptDragPoint.y-point.y;
		m_ptDragPoint=point;

		int nNewPos=m_nScrollPos+nDelta;

		if (nNewPos<0)
			nNewPos=0;
		else if (nNewPos>m_rcOriginalRect.Height() - m_nCurHeight)
			nNewPos=m_rcOriginalRect.Height() - m_nCurHeight;

		nDelta=nNewPos-m_nScrollPos;
		m_nScrollPos=nNewPos;

		SetScrollPos(SB_VERT,m_nScrollPos,TRUE);
		ScrollWindow(0,-nDelta);
	}

	CDialog::OnMouseMove(nFlags, point);
}

void CDlgScrollable::OnKillFocus(CWnd* pNewWnd)
{
	CDialog::OnKillFocus(pNewWnd);

	EndDrag();
}

void CDlgScrollable::EndDrag()
{
	m_bDragging=FALSE;
	ReleaseCapture();
```

```
	SetCursor(m_hCursor1);
}

BOOL CDlgScrollable::OnSetCursor(CWnd* pWnd, UINT nHitTest, UINT message)
{
	BOOL bRet=TRUE;

	if (nHitTest==HTCLIENT)
	{
		SetCursor(m_hCursor1);
		bRet=FALSE;
	}
	else
		bRet=CDialog::OnSetCursor(pWnd,nHitTest,message);

	return bRet;
}
```

Appendix: The Code

DlgScrollable.h

```
#if !defined(AFX_DLGSCROLLABLE_H__3586FDB6_FC0D_4FF7_BA09_D86692F0006A__INCLUDED_)
#define AFX_DLGSCROLLABLE_H__3586FDB6_FC0D_4FF7_BA09_D86692F0006A__INCLUDED_

#if _MSC_VER > 1000
#pragma once
#endif // _MSC_VER > 1000
// DlgScrollable.h : header file
//

/////////////////////////////////////////////////////////////////////////////
// CDlgScrollable dialog

class CDlgScrollable : public CDialog
{
// Construction
public:
	void EndDrag();
	CDlgScrollable(CWnd* pParent = NULL);   // standard constructor

	HCURSOR m_hCursor1, m_hCursor2;

	// dialog size as you see in the resource view (original size)
	CRect	m_rcOriginalRect;
```

```
		// dragging
		BOOL	m_bDragging;
		CPoint	m_ptDragPoint;

		// actual scroll position
		int			m_nScrollPos;

		// actual dialog height
		int			m_nCurHeight;

// Dialog Data
		//{{AFX_DATA(CDlgScrollable)
		enum { IDD = IDD_SCROLLABLE };
			// NOTE: the ClassWizard will add data members here
		//}}AFX_DATA

// Overrides
		// ClassWizard generated virtual function overrides
		//{{AFX_VIRTUAL(CDlgScrollable)
		protected:
		virtual void DoDataExchange(CDataExchange* pDX);	// DDX/DDV support
		//}}AFX_VIRTUAL

// Implementation
protected:

		// Generated message map functions
		//{{AFX_MSG(CDlgScrollable)
		virtual BOOL OnInitDialog();
		afx_msg void OnVScroll(UINT nSBCode, UINT nPos, CScrollBar* pScrollBar);
		afx_msg void OnSize(UINT nType, int cx, int cy);
		afx_msg BOOL OnMouseWheel(UINT nFlags, short zDelta, CPoint pt);
		virtual void OnCancel();
		virtual void OnOK();
		afx_msg void OnLButtonDown(UINT nFlags, CPoint point);
		afx_msg void OnLButtonUp(UINT nFlags, CPoint point);
		afx_msg void OnMouseMove(UINT nFlags, CPoint point);
		afx_msg void OnKillFocus(CWnd* pNewWnd);
		//}}AFX_MSG
		DECLARE_MESSAGE_MAP()
public:
		afx_msg BOOL OnSetCursor(CWnd* pWnd, UINT nHitTest, UINT message);
};

//{{AFX_INSERT_LOCATION}}
// Microsoft Visual C++ will insert additional declarations immediately before the previous line.

#endif // !defined(AFX_DLGSCROLLABLE_H__3586FDB6_FC0D_4FF7_BA09_D86692F0006A__INCLUDED_)
```

DlgScrollable.cpp

```
// DlgScrollable.cpp : implementation file
//

#include "stdafx.h"
#include "P01.h"
#include "DlgScrollable.h"

#ifdef _DEBUG
#define new DEBUG_NEW
#undef THIS_FILE
static char THIS_FILE[] = __FILE__;
#endif

/////////////////////////////////////////////////////////////////////////////
// CDlgScrollable dialog

CDlgScrollable::CDlgScrollable(CWnd* pParent /*=NULL*/)
	: CDialog(CDlgScrollable::IDD, pParent)
{
	//{{AFX_DATA_INIT(CDlgScrollable)
		// NOTE: the ClassWizard will add member initialization here
	//}}AFX_DATA_INIT

	m_bDragging=FALSE;

	// modeless dialog - don't forget to force the
	// WS_CHILD style in the resource template and remove
	// caption and system menu
	Create(CDlgScrollable::IDD,pParent);
}

void CDlgScrollable::DoDataExchange(CDataExchange* pDX)
{
	CDialog::DoDataExchange(pDX);
	//{{AFX_DATA_MAP(CDlgScrollable)
		// NOTE: the ClassWizard will add DDX and DDV calls here
	//}}AFX_DATA_MAP
}

BEGIN_MESSAGE_MAP(CDlgScrollable, CDialog)
	//{{AFX_MSG_MAP(CDlgScrollable)
	ON_WM_VSCROLL()
	ON_WM_SIZE()
	ON_WM_MOUSEWHEEL()
	ON_WM_LBUTTONDOWN()
	ON_WM_LBUTTONUP()
	ON_WM_MOUSEMOVE()
	ON_WM_KILLFOCUS()
	//}}AFX_MSG_MAP
	ON_WM_SETCURSOR()
END_MESSAGE_MAP()
```

```
/////////////////////////////////////////////////////////////////////////////
// CDlgScrollable message handlers

BOOL CDlgScrollable::OnInitDialog()
{
	CDialog::OnInitDialog();

	m_hCursor1=AfxGetApp()->LoadCursor(IDC_CURSOR1);
	m_hCursor2=AfxGetApp()->LoadCursor(IDC_CURSOR2);

	//SetClassLong(m_hWnd,GCL_HCURSOR,(long)m_hCursor1);

	// save the original size
	GetWindowRect(m_rcOriginalRect);

	// initial scroll position
	m_nScrollPos = 0;

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

void CDlgScrollable::OnVScroll(UINT nSBCode, UINT nPos, CScrollBar* pScrollBar)
{
	int nDelta;
	int nMaxPos = m_rcOriginalRect.Height() - m_nCurHeight;

	switch (nSBCode)
	{
	case SB_LINEDOWN:
		if (m_nScrollPos >= nMaxPos)
			return;

		nDelta = min(max(nMaxPos/20,5),nMaxPos-m_nScrollPos);
		break;

	case SB_LINEUP:
		if (m_nScrollPos <= 0)
			return;
		nDelta = -min(max(nMaxPos/20,5),m_nScrollPos);
		break;
	case SB_PAGEDOWN:
		if (m_nScrollPos >= nMaxPos)
			return;
		nDelta = min(max(nMaxPos/10,5),nMaxPos-m_nScrollPos);
		break;
	case SB_THUMBTRACK:
	case SB_THUMBPOSITION:
		nDelta = (int)nPos - m_nScrollPos;
		break;

	case SB_PAGEUP:
		if (m_nScrollPos <= 0)
```

```
                        return;
                nDelta = -min(max(nMaxPos/10,5),m_nScrollPos);
                break;

        default:
                return;
        }
        m_nScrollPos += nDelta;
        SetScrollPos(SB_VERT,m_nScrollPos,TRUE);
        ScrollWindow(0,-nDelta);
        CDialog::OnVScroll(nSBCode, nPos, pScrollBar);
}

void CDlgScrollable::OnSize(UINT nType, int cx, int cy)
{
        CDialog::OnSize(nType, cx, cy);

        m_nCurHeight = cy;

        SCROLLINFO si;
        si.cbSize = sizeof(SCROLLINFO);
        si.fMask = SIF_ALL;
        si.nMin = 0;
        si.nMax = m_rcOriginalRect.Height();
        si.nPage = cy;
        si.nPos = 0;
   SetScrollInfo(SB_VERT, &si, TRUE);

}

BOOL CDlgScrollable::OnMouseWheel(UINT nFlags, short zDelta, CPoint pt)
{
        int nMaxPos = m_rcOriginalRect.Height() - m_nCurHeight;

        if (zDelta<0)
        {
                if (m_nScrollPos < nMaxPos)
                {
                        zDelta = min(max(nMaxPos/20,5),nMaxPos-m_nScrollPos);

                        m_nScrollPos += zDelta;
                        SetScrollPos(SB_VERT,m_nScrollPos,TRUE);
                        ScrollWindow(0,-zDelta);
                }
        }
        else
        {
                if (m_nScrollPos > 0)
                {
                        zDelta = -min(max(nMaxPos/20,5),m_nScrollPos);

                        m_nScrollPos += zDelta;
                        SetScrollPos(SB_VERT,m_nScrollPos,TRUE);
```

```
                        ScrollWindow(0,-zDelta);
                }
        }

        return CDialog::OnMouseWheel(nFlags, zDelta, pt);
}

void CDlgScrollable::OnCancel()
{
}

void CDlgScrollable::OnOK()
{
}

void CDlgScrollable::OnLButtonDown(UINT nFlags, CPoint point)
{
        m_bDragging=TRUE;
        SetCapture();

        m_ptDragPoint=point;

        SetCursor(m_hCursor2);

        CDialog::OnLButtonDown(nFlags, point);
}

void CDlgScrollable::OnLButtonUp(UINT nFlags, CPoint point)
{
        EndDrag();

        CDialog::OnLButtonUp(nFlags, point);
}

void CDlgScrollable::OnMouseMove(UINT nFlags, CPoint point)
{
        if (m_bDragging)
        {
                int nDelta=m_ptDragPoint.y-point.y;
                m_ptDragPoint=point;

                int nNewPos=m_nScrollPos+nDelta;

                if (nNewPos<0)
                        nNewPos=0;
                else if (nNewPos>m_rcOriginalRect.Height() - m_nCurHeight)
                        nNewPos=m_rcOriginalRect.Height() - m_nCurHeight;

                nDelta=nNewPos-m_nScrollPos;
                m_nScrollPos=nNewPos;

                SetScrollPos(SB_VERT,m_nScrollPos,TRUE);
                ScrollWindow(0,-nDelta);
```

```
		}

		CDialog::OnMouseMove(nFlags, point);
}

void CDlgScrollable::OnKillFocus(CWnd* pNewWnd)
{
		CDialog::OnKillFocus(pNewWnd);

		EndDrag();
}

void CDlgScrollable::EndDrag()
{
		m_bDragging=FALSE;
		ReleaseCapture();
		SetCursor(m_hCursor1);
}

BOOL CDlgScrollable::OnSetCursor(CWnd* pWnd, UINT nHitTest, UINT message)
{
		BOOL bRet=TRUE;

		if (nHitTest==HTCLIENT)
		{
			SetCursor(m_hCursor1);
			bRet=FALSE;
		}
		else
			bRet=CDialog::OnSetCursor(pWnd,nHitTest,message);

		return bRet;
}
```