CAGD Kit Reference: demonstration example.

Module: demo.c

```
#include < cagd.h >
#include < stdio.h >
#include "resource.h"


enum {
        MY_CLICK = CAGD_USER,
        MY_POLY,
        MY_ANIM,
        MY_DRAG,
        MY_ADD,
        MY_COLOR,
        MY_REMOVE,
};


char *animText[] = {
        "Animation Demo",
        "During the animation you can freely\n"
        "rotate, translate and scale the scene."
};

char *dragText[] = {
        "Drag, Popups & Dialog Demo",
        "Click right mouse button to call popup menu.\n"
        "A contents of the menu depends on where you\n"
        "clicked. Clicking on an existing point allows\n"
        "you to remove the point or change its color via\n"
        "dialog. If there is no point you could add one.\n\n"
        "Using left mouse button you can drag an existing\n"
        "points. Watch carefully a text appearing near\n"
        "the point being draged and don't miss."
};

char *clickText[] = {
        "Click Demo",
        "CAGD unable to convert 2D point defined by your\n"
        "click to single 3D point located in the object\n"
        "space. Instead it returns you two 3D points\n"
        "as specification of vector. Initially you are\n"
        "looking in the direction of this vector and cannot\n"
        "see anything. Try to rotate the scene after few\n"
        "clicks. You'll see polylines defined by the returned\n"
        "vectors."
};

char *polyText[] = {
        "Polyline Demo",
        "Click polyline. The nearest vertex will be marked with\n"
        "text. Remember that \"nearest\" defined by the minimal\n"
        "distance on the screen (window coordinates) and not\n"
        "in the object space."
};


HMENU myPopup;
UINT myText;
char myBuffer[BUFSIZ];


/* Using Win32 API to create dialog boxes. */


void myMessage(PSTR title, PSTR message, UINT type)
{
        MessageBox(cagdGetWindow(), message, title, MB_OK | MB_APPLMODAL | type);
}


LRESULT CALLBACK myDialogProc(HWND hDialog, UINT message, WPARAM wParam, LPARAM lParam)
{
        if(message != WM_COMMAND)
                return FALSE;
        switch(LOWORD(wParam)){
        case IDOK:
            GetDlgItemText(hDialog, IDC_EDIT, myBuffer, sizeof(myBuffer));
            EndDialog(hDialog, TRUE);
            return TRUE;
        case IDCANCEL:
            EndDialog(hDialog, FALSE);
            return TRUE;
        default:
            return FALSE;
        }
}


/* CAGD message handlers. */


void myTimer(int x, int y, PVOID userData)
{
        cagdRotate(2, 0, 1, 0);
        cagdRedraw();
}


void myDragRightUp(int x, int y, PVOID userData)
{
        UINT id;
        CAGD_POINT p[2];
        int red, green, blue;
        HMENU hMenu = (HMENU)userData;
        cagdHideSegment(myText);
        for(cagdPick(x, y); id = cagdPickNext();)
                if(cagdGetSegmentType(id) == CAGD_SEGMENT_POINT)
                        break;
        EnableMenuItem(hMenu, MY_ADD, id? MF_GRAYED: MF_ENABLED);
        EnableMenuItem(hMenu, MY_COLOR, id? MF_ENABLED: MF_GRAYED);
        EnableMenuItem(hMenu, MY_REMOVE, id? MF_ENABLED: MF_GRAYED);
        switch(cagdPostMenu(hMenu, x, y)){
        case MY_ADD:
            cagdToObject(x, y, p);
            p->z = 0;
            cagdAddPoint(p);
            break;
        case MY_COLOR:
            if(DialogBox(cagdGetModule(),
                         MAKEINTRESOURCE(IDD_COLOR),
                         cagdGetWindow(),
                         (DLGPROC)myDialogProc))
                if(sscanf(myBuffer, "%d %d %d", &red, &green, &blue) == 3)
                    cagdSetSegmentColor(id, (BYTE)red, (BYTE)green, (BYTE)blue);
                else
                    myMessage("Change color", "Bad color!", MB_ICONERROR);
            break;
        case MY_REMOVE:
            cagdFreeSegment(id);
            break;
        }
        cagdRedraw();
}


void myDragMove(int x, int y, PVOID userData)
{
        CAGD_POINT p[2];
        cagdToObject(x, y, p);
        p->z = 0;
        cagdReusePoint((UINT)userData, p);
        cagdReuseText(myText, p, " Leave me alone!");
        cagdRedraw();
}


void myDragLeftUp(int x, int y, PVOID userData)
{
        CAGD_POINT p;
        cagdGetSegmentLocation(myText, &p);
        cagdReuseText(myText, &p, " Ufff!");
        cagdRegisterCallback(CAGD_MOUSEMOVE, NULL, NULL);
        cagdRegisterCallback(CAGD_LBUTTONUP, NULL, NULL);
        cagdRedraw();
}


void myDragLeftDown(int x, int y, PVOID userData)
{
        UINT id;
        for(cagdPick(x, y); id = cagdPickNext();)
            if(cagdGetSegmentType(id) == CAGD_SEGMENT_POINT)`
                break;
        if(id){
            CAGD_POINT p;
            BYTE red, green, blue;
            cagdGetSegmentLocation(id, &p);
            cagdReuseText(myText, &p, " Don't touch me!");
            cagdGetSegmentColor(id, &red, &green, &blue);
            cagdSetSegmentColor(myText, red, green, blue);
            cagdShowSegment(myText);
            cagdRegisterCallback(CAGD_MOUSEMOVE, myDragMove, (PVOID)id);
            cagdRegisterCallback(CAGD_LBUTTONUP, myDragLeftUp, NULL);
        } else
            myMessage("Ha-ha!", "You missed...", MB_ICONERROR);
        cagdRedraw();
}


void myClickLeftDown(int x, int y, PVOID userData)
{
        CAGD_POINT p[2];
        cagdToObject(x, y, p);
        cagdAddPolyline(p, sizeof(p) / sizeof(CAGD_POINT));
}


void myPolyLeftDown(int x, int y, PVOID userData)
{
        CAGD_POINT p;
        UINT id;
        int v;
        for(cagdPick(x, y); id = cagdPickNext();)
            if(cagdGetSegmentType(id) == CAGD_SEGMENT_POLYLINE)
                break;
        if(id){
            if(v = cagdGetNearestVertex(id, x, y)){
                cagdGetVertex(id, --v, &p);
                sprintf(myBuffer, " near #%d", v);
                cagdReuseText(myText, &p, myBuffer);
                cagdShowSegment(myText);
            }
        } else
            myMessage("Ha-ha!", "You missed...", MB_ICONERROR);
        cagdRedraw();
}


void myCommand(int id, int unUsed, PVOID userData)
{
        int i;
        CAGD_POINT p[] = {{1, 1, 1}, {1, -1, 1}, {-1, -1, 1}, {-1, 1, 1}, {1, 1, 1}};
        static state = MY_CLICK;
        HMENU hMenu = (HMENU)userData;
        CheckMenuItem(hMenu, state, MF_UNCHECKED);
        CheckMenuItem(hMenu, state = id, MF_CHECKED);
        cagdFreeAllSegments();
        cagdReset();
        cagdSetView(CAGD_ORTHO);
        cagdSetDepthCue(FALSE);
        cagdSetColor(255, 255, 255);
        cagdRegisterCallback(CAGD_TIMER, NULL, NULL);
        cagdRegisterCallback(CAGD_LBUTTONDOWN, NULL, NULL);
        cagdRegisterCallback(CAGD_LBUTTONUP, NULL, NULL);
        cagdRegisterCallback(CAGD_RBUTTONUP, NULL, NULL);
        cagdRedraw();
        switch(id){
        case MY_ANIM:
            cagdSetView(CAGD_PERSP);
            cagdSetDepthCue(TRUE);
            cagdSetColor(0, 255, 255);
            cagdAddPolyline(p, sizeof(p) / sizeof(CAGD_POINT));
            p[0].z = p[1].z = p[2].z = p[3].z = p[4].z = -1;
            cagdAddPolyline(p, sizeof(p) / sizeof(CAGD_POINT));
            p[1].z = p[2].z = p[1].y = p[2].y = 1;
            cagdAddPolyline(p, sizeof(p) / sizeof(CAGD_POINT));
            p[0].y = p[1].y = p[2].y = p[3].y = p[4].y = -1;
            cagdAddPolyline(p, sizeof(p) / sizeof(CAGD_POINT));
            for(i = 0; i < 100; i++){
                p->x = (GLdouble)rand() / RAND_MAX * 2 - 1;
                p->y = (GLdouble)rand() / RAND_MAX * 2 - 1;
                p->z = (GLdouble)rand() / RAND_MAX * 2 - 1;
                cagdAddPoint(p);
            }
            SetWindowText(cagdGetWindow(), animText[0]);
            cagdSetHelpText(animText[1]);
            cagdShowHelp();
            cagdRegisterCallback(CAGD_TIMER, myTimer, NULL);
            break;
        case MY_DRAG:
            SetWindowText(cagdGetWindow(), dragText[0]);
            cagdSetHelpText(dragText[1]);
            cagdShowHelp();
            cagdHideSegment(myText = cagdAddText(p, ""));
            cagdRegisterCallback(CAGD_RBUTTONUP, myDragRightUp, (PVOID)myPopup);
            cagdRegisterCallback(CAGD_LBUTTONDOWN, myDragLeftDown, NULL);
            break;
        case MY_CLICK:
            cagdSetView(CAGD_PERSP);
            cagdSetDepthCue(TRUE);
            SetWindowText(cagdGetWindow(), clickText[0]);
            cagdSetHelpText(clickText[1]);
            cagdShowHelp();
            cagdRegisterCallback(CAGD_LBUTTONDOWN, myClickLeftDown, NULL);
            break;
        case MY_POLY:
            cagdScale(0.5, 0.5, 0.5);
            cagdRotate(45, 0, 0, 1);
            cagdAddPolyline(p, sizeof(p) / sizeof(CAGD_POINT) - 1);
            cagdHideSegment(myText = cagdAddText(p, ""));
            SetWindowText(cagdGetWindow(), polyText[0]);
            cagdSetHelpText(polyText[1]);
            cagdShowHelp();
            cagdRegisterCallback(CAGD_LBUTTONDOWN, myPolyLeftDown, NULL);
            break;
        }
        cagdRedraw();
}


int main(int argc, char *argv[])
{
        HMENU hMenu;
        cagdBegin("CAGD", 512, 512);
        hMenu = CreatePopupMenu();
        AppendMenu(hMenu, MF_STRING, MY_CLICK, "Click");
        AppendMenu(hMenu, MF_STRING, MY_POLY, "Polyline");
        AppendMenu(hMenu, MF_STRING, MY_ANIM, "Animation");
        AppendMenu(hMenu, MF_STRING, MY_DRAG, "Drag, Popup & Dialog");
        cagdAppendMenu(hMenu, "Demos");
        myPopup = CreatePopupMenu();
        AppendMenu(myPopup, MF_STRING | MF_DISABLED, 0, "Point");
        AppendMenu(myPopup, MF_SEPARATOR, 0, NULL);
        AppendMenu(myPopup, MF_STRING, MY_ADD, "Add");
        AppendMenu(myPopup, MF_SEPARATOR, 0, NULL);
        AppendMenu(myPopup, MF_STRING, MY_COLOR, "Change color...");
        AppendMenu(myPopup, MF_STRING, MY_REMOVE, "Remove");
        cagdRegisterCallback(CAGD_MENU, myCommand, (PVOID)hMenu);
        cagdShowHelp();
        cagdMainLoop();
        return 0;
}
```

Module: demo.rc

```
//Microsoft Developer Studio generated resource script.
//
#include "resource.h"

#define APSTUDIO_READONLY_SYMBOLS
/////////////////////////////////////////////////////////////////////////////
//
// Generated from the TEXTINCLUDE 2 resource.
//
#include "afxres.h"

/////////////////////////////////////////////////////////////////////////////
#undef APSTUDIO_READONLY_SYMBOLS

/////////////////////////////////////////////////////////////////////////////
// English (U.S.) resources

#if !defined(AFX_RESOURCE_DLL) || defined(AFX_TARG_ENU)
#ifdef _WIN32
LANGUAGE LANG_ENGLISH, SUBLANG_ENGLISH_US
#pragma code_page(1252)
#endif //_WIN32

/////////////////////////////////////////////////////////////////////////////
//
// Dialog
//

IDD_COLOR DIALOG DISCARDABLE  0, 0, 183, 70
STYLE DS_MODALFRAME | DS_3DLOOK | WS_POPUP | WS_CAPTION | WS_SYSMENU
CAPTION "Change color"
FONT 8, "MS Sans Serif"
BEGIN
    DEFPUSHBUTTON   "OK",IDOK,25,48,50,14
    PUSHBUTTON      "Cancel",IDCANCEL,107,48,50,14
    EDITTEXT        IDC_EDIT,7,23,169,14,ES_AUTOHSCROLL
    CTEXT           "E.g. ""255 255 255""",IDC_STATIC,7,8,169,8
END


/////////////////////////////////////////////////////////////////////////////
//
// DESIGNINFO
//

#ifdef APSTUDIO_INVOKED
GUIDELINES DESIGNINFO DISCARDABLE
BEGIN
    IDD_COLOR, DIALOG
    BEGIN
        LEFTMARGIN, 7
        RIGHTMARGIN, 176
        TOPMARGIN, 8
        BOTTOMMARGIN, 62
    END
END
#endif    // APSTUDIO_INVOKED


#ifdef APSTUDIO_INVOKED
/////////////////////////////////////////////////////////////////////////////
//
// TEXTINCLUDE
//

1 TEXTINCLUDE DISCARDABLE
BEGIN
    "resource.h\0"
END

2 TEXTINCLUDE DISCARDABLE
BEGIN
    "#include ""afxres.h""\r\n"
    "\0"
END

3 TEXTINCLUDE DISCARDABLE
BEGIN
    "\r\n"
    "\0"
END

#endif    // APSTUDIO_INVOKED

#endif    // English (U.S.) resources
/////////////////////////////////////////////////////////////////////////////



#ifndef APSTUDIO_INVOKED
/////////////////////////////////////////////////////////////////////////////
//
// Generated from the TEXTINCLUDE 3 resource.
//


/////////////////////////////////////////////////////////////////////////////
#endif    // not APSTUDIO_INVOKED
```

Module: resource.h

```
//{{NO_DEPENDENCIES}}
// Microsoft Developer Studio generated include file.
// Used by demo.rc
//
#define IDD_COLOR                       101
#define IDC_EDIT                        1005

// Next default values for new objects
//
#ifdef APSTUDIO_INVOKED
#ifndef APSTUDIO_READONLY_SYMBOLS
#define _APS_NEXT_RESOURCE_VALUE        103
#define _APS_NEXT_COMMAND_VALUE         40003
#define _APS_NEXT_CONTROL_VALUE         1007
#define _APS_NEXT_SYMED_VALUE           101
#endif
#endif
```