

Ứng dụng MFC (Visual C++) trong mô phỏng Robot và hệ Cơ điện tử

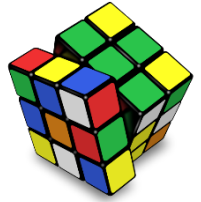


Bài 10: Thiết kế nâng cao với ứng dụng kiểu SDI

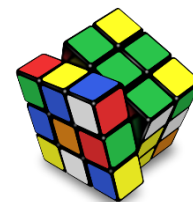
PHẠM MINH QUÂN

mquan.ph@gmail.com

Nội dung



1. Ứng dụng SDI với kiến trúc Document/View
2. Thêm các cửa sổ chức năng
3. Tương tác từ các cửa sổ tới khung View
4. Tương tác từ View tới các cửa sổ khác



1. Ứng dụng SDI với kiến trúc Doc/View

❑ Tạo project mới: ứng dụng MFC, kiểu SDI, kiến trúc Document/ View

MFC Application Wizard - MFCmyApp

Application Type

Overview
Application Type
Compound Document Support
Document Template Properties
Database Support
User Interface Features
Advanced Features
Generated Classes

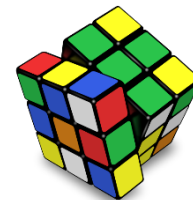
Application type:
☒ Single document
☐ Multiple documents
☐ Tabbed documents
☐ Dialog based
☐ Use HTML dialog
☐ No enhanced MFC controls
☐ Multiple top-level documents
☒ Document/view architecture support
☒ Security Development Lifecycle (SDL) checks
Resource language:
English (United States)
☒ Use Unicode libraries

Project style:
☐ MFC standard
☐ Windows Explorer
☒ Visual Studio
☐ Office

Visual style and colors:
Visual Studio 2008
☐ Enable visual style switching

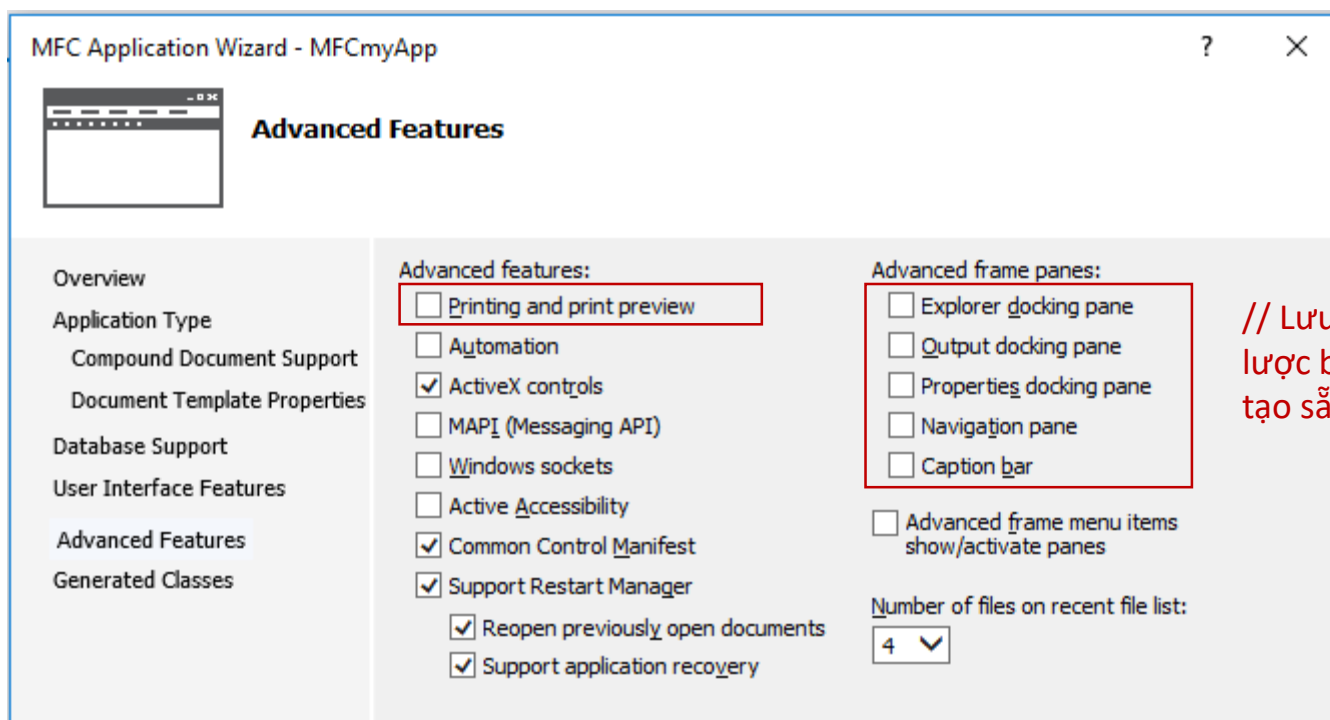
Use of MFC:
☒ Use MFC in a shared DLL
☐ Use MFC in a static library

// Bỏ chọn để lược bớt nội dung tạo sẵn không cần thiết

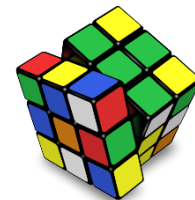


1. Ứng dụng SDI với kiến trúc Doc/View

❑ Tạo project mới: ứng dụng MFC, kiểu SDI, kiến trúc Document/ View



// Lưu ý: bỏ chọn để lược bớt các nội dung tạo sẵn không cần thiết



1. Ứng dụng SDI với kiến trúc Doc/View

❑ Cài đặt OpenGL cho khung View

- Thực hiện tương tự như cài đặt OpenGL vào khung ChildView đã trình bày trong Bài 9 (Thay vì thao tác với lớp CChildView, ta thao tác với lớp C...View)

// File "...View.h"

```
#include "OpenGLControl.h"

...

COpenGLControl m_oglWindow;

...
```

// File "...View.cpp"

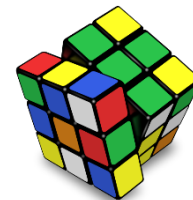
```
void CMFCMyAppView::OnPaint()
{
    CPaintDC dc(this); // device context for painting
    // TODO: Add your message handler code here
    // Do not call CView::OnPaint() for painting messages
    m_oglWindow.oglDrawScene();
}

int CMFCMyAppView::OnCreate(LPCREATESTRUCT lpCreateStruct)
{
    if (CView::OnCreate(lpCreateStruct) == -1) return -1;

    // TODO: Add your specialized creation code here
    CRect rect;
    GetWindowRect(rect);
    ScreenToClient(rect);
    m_oglWindow.oglCreate(rect, this);

    return 0;
}

void CMFCMyAppView::OnSize(UINT nType, int cx, int cy)
{
    CView::OnSize(nType, cx, cy);
    // TODO: Add your message handler code here
    m_oglWindow.MoveWindow(0, 0, cx, cy);
}
```



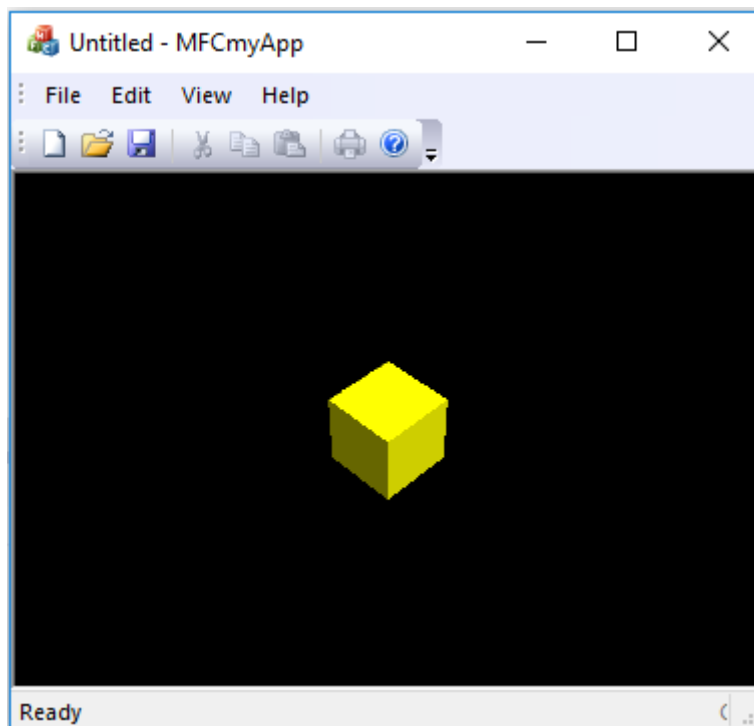
1. Ứng dụng SDI với kiến trúc Doc/View

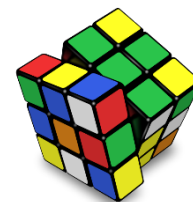
☐ Cài đặt OpenGL cho khung View

Lưu ý: Để tránh xung đột với thao tác chuột của lớp COpenGLControl, ta tạm thời xóa nội dung của hàm OnContextMenu() của lớp C...View



Kết quả chạy:



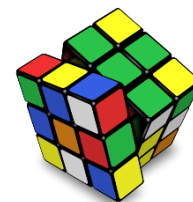


2. Thêm các cửa sổ chức năng

❑ Thêm một cửa sổ CDockablePane gắn vào MainFrame

➤ Tạo một lớp mới dẫn xuất từ lớp CDockablePane

The image shows two overlapping windows from the Visual Studio IDE. The left window is titled 'Add Class - MFCmyApp'. It has a tree view on the left with 'Visual C++' expanded, showing 'CLR', 'ATL', 'C++', and 'MFC'. The 'MFC' item is selected and highlighted with a red box and labeled '//1.'. The main area of this window shows a list of MFC class templates: 'MFC Class' (highlighted with a red box and labeled '//2.'), 'MFC Class From TypeLib', 'MFC Class From ActiveX Control', and 'MFC ODBC Consumer'. The right window is titled 'MFC Add Class Wizard - MFCmyApp'. It has a 'Names' tab selected. The 'Class name' field is 'ActionPane' (labeled '//3.'). The 'Base class' dropdown is set to 'CDockablePane' (labeled '//4.'). Other fields include 'Dialog ID' (IDD_ACTIONPANE), '.h file' (ActionPane.h), and '.cpp file' (ActionPane.cpp). There is an 'Active accessibility' checkbox at the bottom.



2. Thêm các cửa sổ chức năng

❑ Thêm một cửa sổ CDockablePane gắn vào MainFrame

➤ Khai báo và sử dụng đối tượng của lớp mới trong lớp MainFrame

// File MainFrm.h

```
#include "ActionPane.h"

...

ActionPane m_ActPane;

...
```

// File MainFrm.cpp

```
int CMainFrame::OnCreate(LPCREATESTRUCT lpCreateStruct)
{
    ...

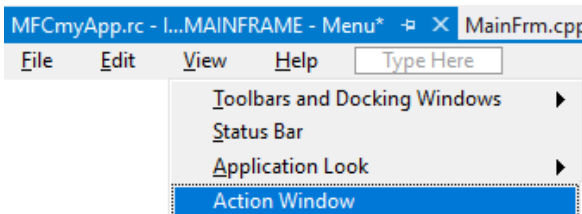
    if (!m_ActPane.Create(_T("Action Window"), this,
        CRect(0, 0, 300, 300), TRUE, ID_VIEW_ACTIONWINDOW,
        WS_CHILD | WS_VISIBLE | WS_CLIPSIBLINGS |
        WS_CLIPCHILDREN | CBRS_LEFT | CBRS_FLOAT_MULTIPLE))
    {
        TRACE0("Failed to create State window\n");
        return FALSE; // failed to create
    }

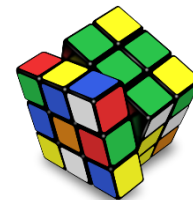
    m_ActPane.EnableDocking(CBRS_LEFT|CBRS_RIGHT);
    DockPane(&m_ActPane);

    ...
}
```

// Lưu ý:

- Tạo một ID bất kỳ bằng cách: Resource view -> Chuột phải -> Resource Symbols -> New...
- Hoặc tạo thêm một mục trong Menu IDR_MAINFRAME và copy ID vào dòng lệnh trên (cách thêm menu đã trình bày trong Bài 9).





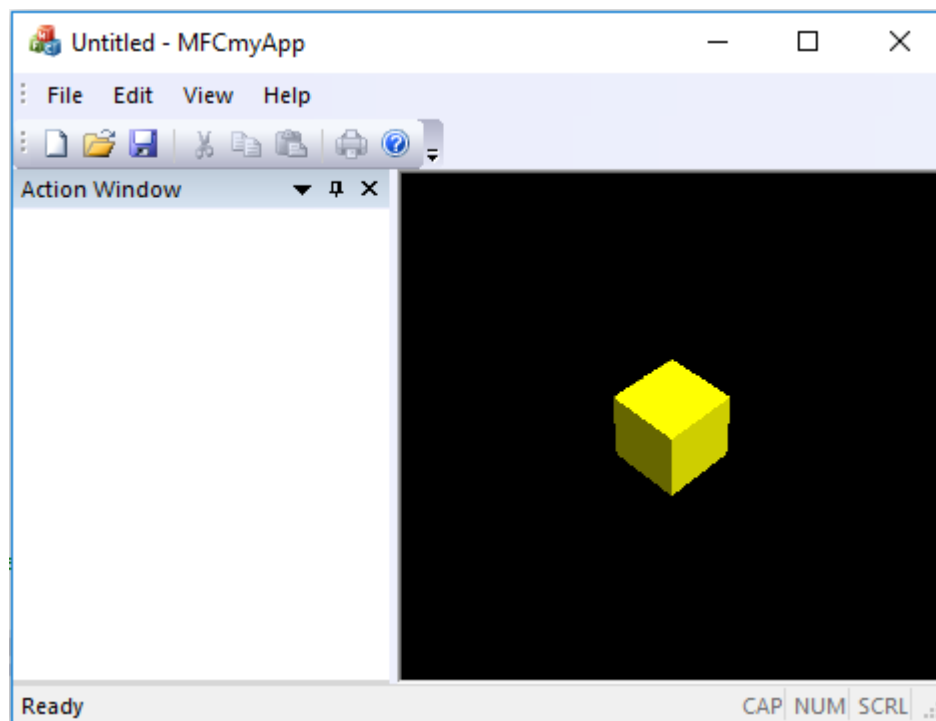
2. Thêm các cửa sổ chức năng

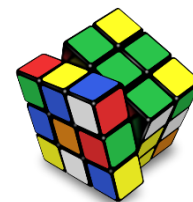
❑ Thêm một cửa sổ CDockablePane gắn vào MainFrame



Kết quả chạy:

- Xuất hiện khung cửa sổ trống bên trái khung view

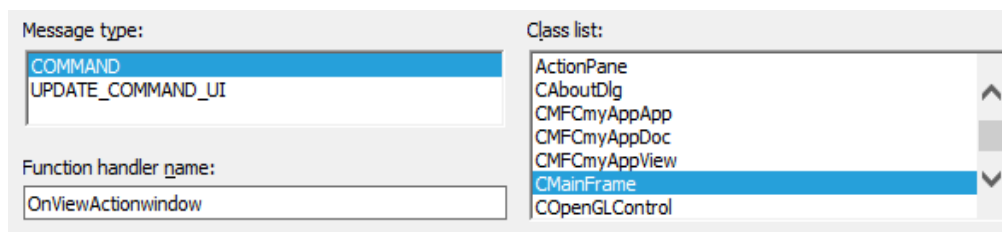




2. Thêm các cửa sổ chức năng

❑ Thêm một cửa sổ CDockablePane gắn vào MainFrame

- Viết code cho nút lệnh vừa thêm trên menu, thực hiện chức năng Hiện/Ẩn ô cửa sổ vừa tạo (cách thêm hàm sự kiện cho nút lệnh đã trình bày trong Bài 9)

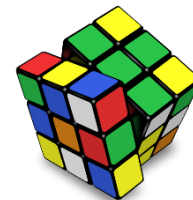


```
void CMainFrame::OnViewActionwindow()
{
    // TODO: Add your command handler code here
    if (m_ActPane.IsPaneVisible()) m_ActPane.ShowPane(FALSE, FALSE, FALSE);
    else
    {
        m_ActPane.ShowPane(TRUE, FALSE, FALSE);
        m_ActPane.SetAutoHideMode(FALSE, CBRS_LEFT);
    }
}
```



Kết quả chạy:

- Khi click vào nút lệnh trên menu, nếu ô cửa đang hiện thì sẽ ẩn đi, nếu ô cửa đang ẩn thì sẽ hiện lên.



2. Thêm các cửa sổ chức năng

❑ Thêm một cửa sổ CDockablePane gắn vào MainFrame

- Để cập nhật trạng thái Hiện/Ẩn của ô cửa sổ vừa tạo trên nút lệnh menu (hiển thị dấu ☒ khi ô cửa đang hiện, bỏ trống khi ô cửa đang ẩn) ta thêm hàm chức năng sau đây cho nút lệnh

Message type:	Class list:
COMMAND	ActionPane
UPDATE_COMMAND_UI	CAboutDlg
	CMFCmyAppApp
	CMFCmyAppDoc
	CMFCmyAppView
	CMainFrame
	COpenGLControl
Function handler name:	
OnUpdateViewActionwindow	

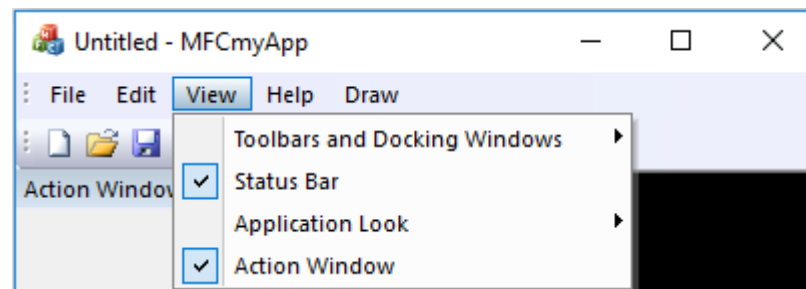


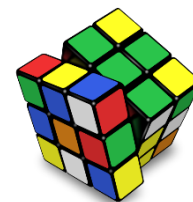
```
void CMainFrame::OnUpdateViewActionwindow(CCmdUI *pCmdUI)
{
    // TODO: Add your command update UI handler code here
    pCmdUI->SetCheck(m_ActPane.IsPaneVisible());
}
```



Kết quả chạy:

- Lệnh trên menu thể hiện trạng thái Hiện/Ẩn của ô cửa thông qua dấu Check ☒

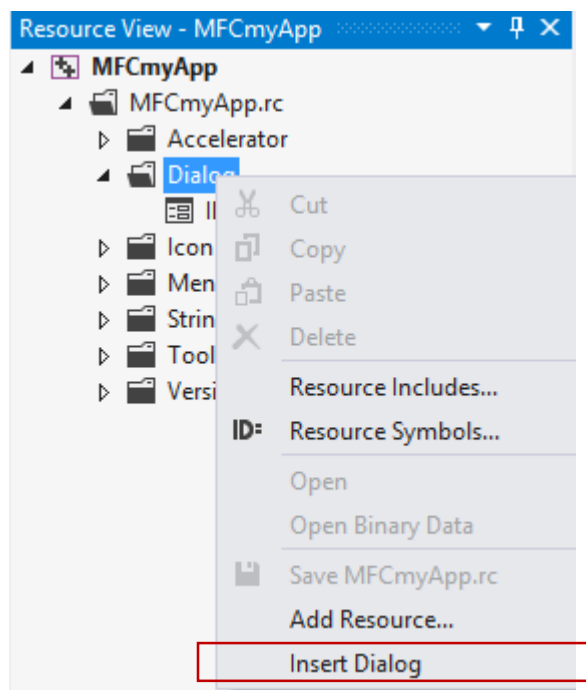




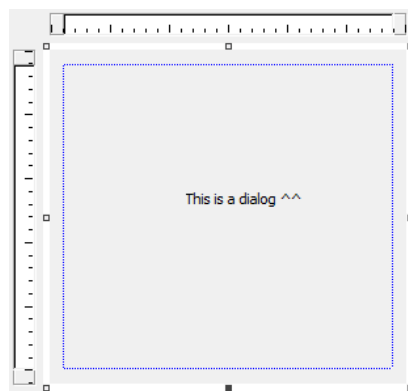
2. Thêm các cửa sổ chức năng

❑ Thêm một Dialog vào bên trong cửa sổ CDockablePane

➤ Thêm Dialog mới bằng Add -> Resource -> Dialog hoặc Insert Dialog

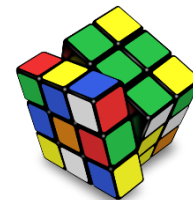


➤ Thay đổi properties của Dialog cho phù hợp để có thể gắn bên trong ô cửa sổ (...pane)



- *Border: None*
- *Style: Child*
- *ID: <tùy ý>*

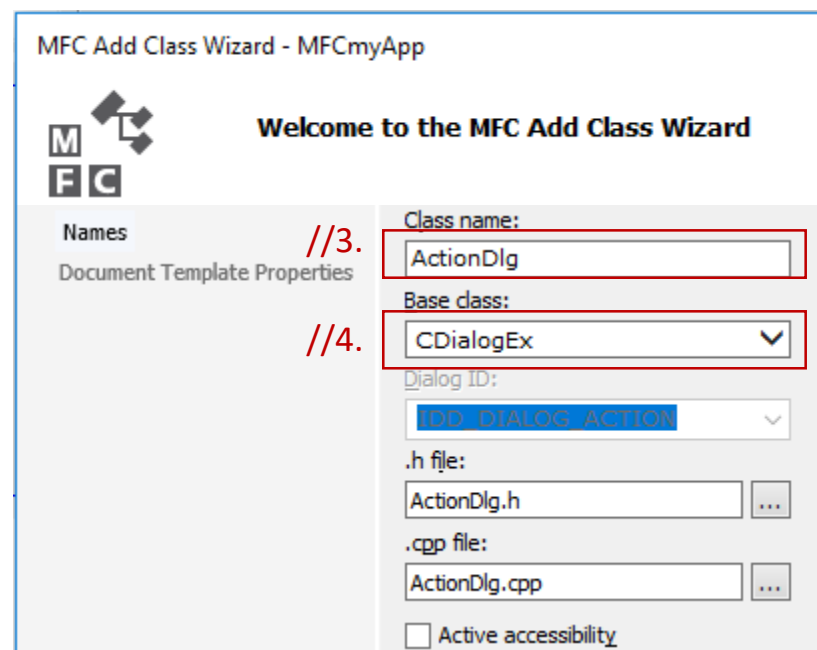
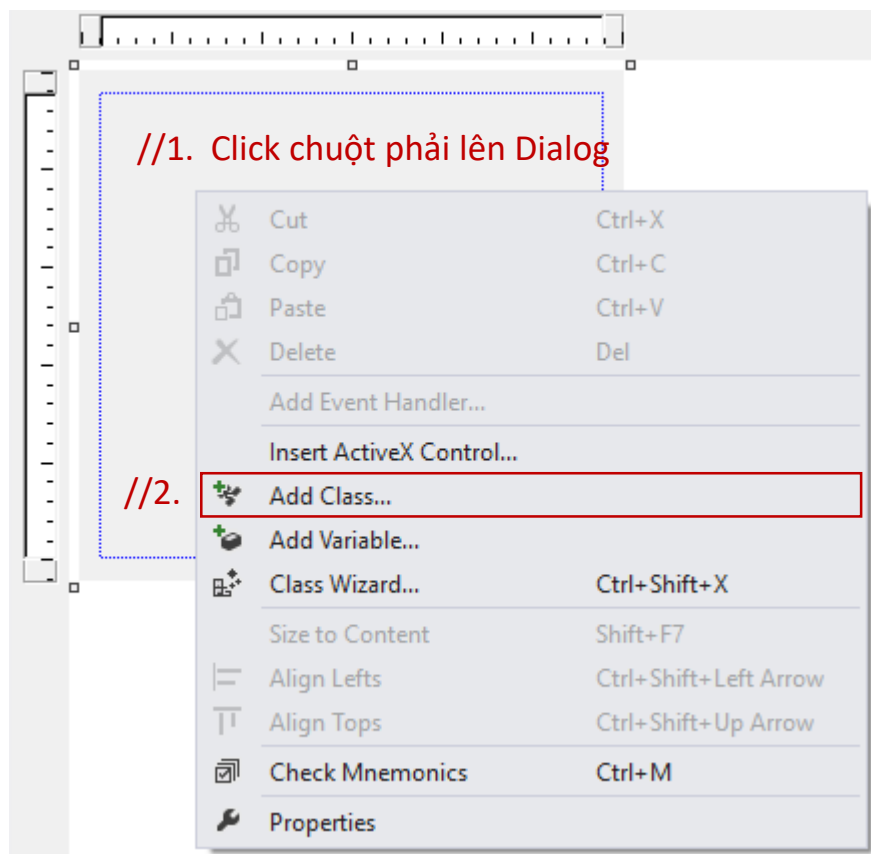
Border	None
Style	Child
ID	IDD_DIALOG_ACTION



2. Thêm các cửa sổ chức năng

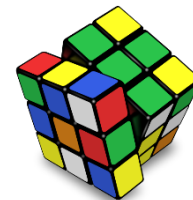
❑ Thêm một Dialog vào bên trong cửa sổ CDockablePane

➤ Thêm lớp quản lý Dialog mới tạo



*Lưu ý: Khai báo thêm trong file *...Dlg.h* dòng lệnh sau:

```
#include "resource.h" //5.
```



2. Thêm các cửa sổ chức năng

❑ Thêm một Dialog vào bên trong cửa sổ CDockablePane

➤ Khai báo và sử dụng đối tượng Dialog bên trong lớp Pane

- Khai báo trong file *...Pane.h*

```
#include "ActionDlg.h"
```

```
    ActionDlg m_ActDlg;
```

- Thêm sự kiện WM_CREATE cho lớp ...Pane
- Viết code hàm OnCreate() của lớp ...Pane như sau

```
int ActionPane::OnCreate(LPCREATESTRUCT lpCreateStruct)
{
    if (CDockablePane::OnCreate(lpCreateStruct) == -1)
        return -1;

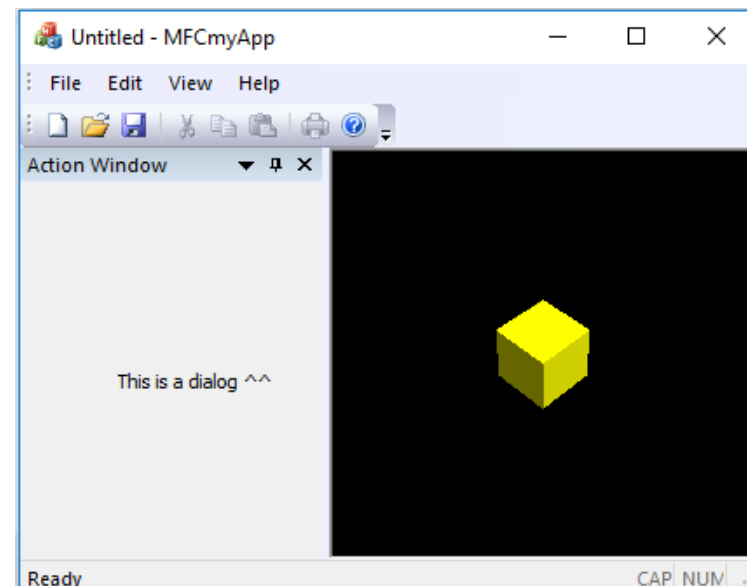
    // TODO: Add your specialized creation code here
    BOOL bRet = m_ActDlg.Create(IDD_DIALOG_ACTION, this);
    ASSERT( bRet );
    m_ActDlg.ShowWindow(SW_SHOW);

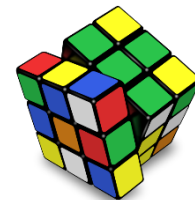
    return 0;
}
```



Kết quả chạy:

- Giao diện Dialog xuất hiện trong khung Pane





3. Tương tác với khung View

❑ Truy cập View từ MainFrame

- Thêm lớp vẽ vật thể OpenGL như trình bày trong Bài 9

▷ OglObject.h

▷ OglObject.cpp

- Thêm lệnh trên Menu và Toolbar có chức năng chọn hình khối vật thể (hình hộp, hình cầu...) như trình bày trong Bài 9.

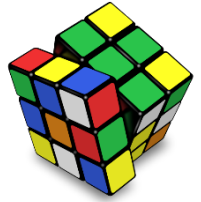
Riêng code cho các hàm sự kiện của các nút lệnh mới thay đổi như sau:

```
#include "MFCmyAppDoc.h"
#include "MFCmyAppView.h"
...
void CMainFrame::OnDrawBox()
{
    // TODO: Add your command handler code here
    CMFCmyAppView *pView = (CMFCmyAppView*)GetActiveView();
    pView->m_oglWindow.Obj1.SetType(BOX);
    pView->m_oglWindow.oglDrawScene();
}
```

```
void CMainFrame::OnDrawSphere()
{
    // TODO: Add your command handler code here
    CMFCmyAppView *pView = (CMFCmyAppView*)GetActiveView();
    pView->m_oglWindow.Obj1.SetType(SPHERE);
    pView->m_oglWindow.oglDrawScene();
}
```



*Kết quả chạy:
- Như mong muốn*



3. Tương tác với khung View

❑ Truy cập View từ các lớp khác

- Thêm hàm GetView() vào lớp C...View

// File ...View.h

```
| static CMFCmyAppView * GetView();
```

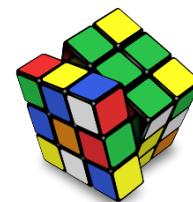
// File ...View.cpp

```
CMFCmyAppView * CMFCmyAppView::GetView()
{
    CFrameWnd * pFrame = (CFrameWnd *) (AfxGetApp()->m_pMainWnd);
    CView * pView = pFrame->GetActiveView();

    if ( !pView )
        return NULL;

    // Fail if view is of wrong kind
    // (this could occur with splitter windows, or additional
    // views on a single document
    if ( ! pView->IsKindOf( RUNTIME_CLASS(CMFCmyAppView) ) )
        return NULL;

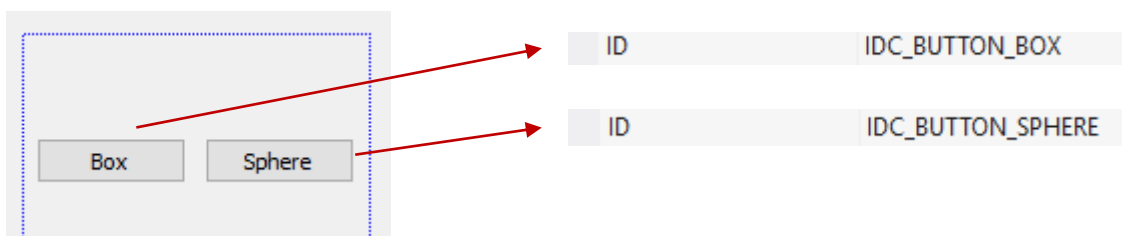
    return (CMFCmyAppView *) pView;
}
```

3. Tương tác với khung View

❑ Truy cập View từ các lớp khác

- Thêm vào Dialog (mới tạo) 2 Button như sau:

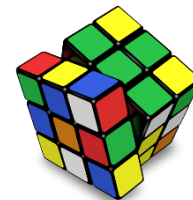


- Nháy đúp vào từng Button để thêm hàm sự kiện OnBnClicked...(), viết code như sau:

```
#include "MFCmyAppDoc.h"
#include "MFCmyAppView.h"
...
// File ...Dlg.cpp

void ActionDlg::OnBnClickedButtonBox()
{
    // TODO: Add your control notification handler code here
    CMFCmyAppView *pView = CMFCmyAppView::GetView();
    pView->m_oglWindow.Obj1.SetType(BOX);
    pView->m_oglWindow.oglDrawScene();
}

void ActionDlg::OnBnClickedButtonSphere()
{
    // TODO: Add your control notification handler code here
    CMFCmyAppView *pView = CMFCmyAppView::GetView();
    pView->m_oglWindow.Obj1.SetType(SPHERE);
    pView->m_oglWindow.oglDrawScene();
}
```



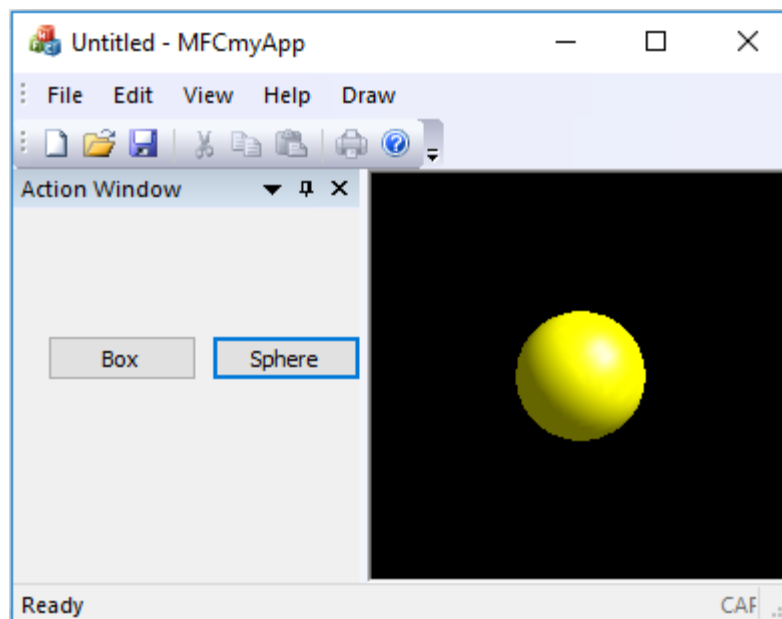
3. Tương tác với khung View

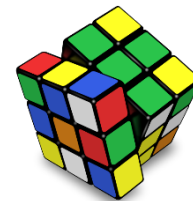
☐ Truy cập View từ các lớp khác



Kết quả chạy:

- Khi nhấn các nút trên Dialog thì hình vẽ OpenGL thay đổi





3. Tương tác với khung View

❑ Sử dụng Slider Control

- Thêm vào Dialog các Slider Control như dưới đây



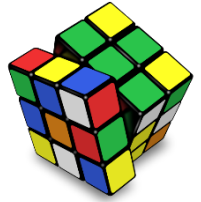
- Thêm biến điều khiển cho các Slider Control (Chuột phải -> Add Variable...)

// File ...Dlg.h

```
CSliderCtrl SliderX_ctrl;  
CSliderCtrl SliderY_ctrl;
```

// File ...Dlg.cpp

```
void ActionDlg::DoDataExchange(CDataExchange* pDX)  
{  
    CDialogEx::DoDataExchange(pDX);  
    DDX_Control(pDX, IDC_SLIDER_X, SliderX_ctrl);  
    DDX_Control(pDX, IDC_SLIDER_Y, SliderY_ctrl);  
}
```



3. Tương tác với khung View

❑ Sử dụng Slider Control

- Thêm vào lớp OglObject các thuộc tính chỉ vị trí theo phương x và y.
Thêm và sửa code của các hàm có liên quan

// File OglObject.h

#pragma once

#define BOX 1

#define SPHERE 2

class OglObject

{

public:

OglObject(void);

~OglObject(void);

protected:

int type;

float x, y;

public:

void SetType(int);

void SetPositionX(float);

void SetPositionY(float);

float GetX();

float GetY();

void Draw();

};

// File OglObject.cpp

OglObject::OglObject(void) {type = BOX; x = 0; y = 0;}

OglObject::~OglObject(void) {}

void OglObject::SetType(int tp) {type = tp;}

void OglObject::SetPositionX(float posx) {x = posx;}

void OglObject::SetPositionY(float posy) {y = posy;}

float OglObject::GetX() {return x;}

float OglObject::GetY() {return y;}

void OglObject::Draw()

{

glPushMatrix();

glTranslatef(x, y, 0);

switch(type)

{

case BOX:

glutSolidCube (1.0);

break;

case SPHERE:

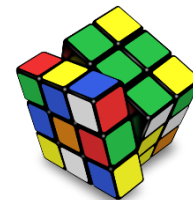
glutSolidSphere(1.0, 30, 30);

break;

}

glPopMatrix();

}



3. Tương tác với khung View

❑ Sử dụng Slider Control

- Thêm sự kiện WM_HSCROLL vào lớp Dialog

Project: MFCmyApp //1. Class name: ActionDlg

Base class: CDialogEx Class declaration: actiondlg.h

Resource: IDD_DIALOG_ACTION Class implementation: actiondlg.cpp

//2. Messages Virtual Functions Member Variables Methods

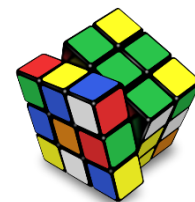
Search Messages Search Handlers //4.

Messages: WM_EXITSZEMOVE WM_FONTCHANGE WM_GETDLGCODE WM_GETMINMAXINFO WM_HELPINFO WM_HOTKEY //3. WM_HSCROLL WM_HSCROLLCLIPBOARD WM_ICONFRASFRKGN

Existing handlers:

Function name	Message
OnHScroll	WM_HSCROLL

Add Handler Delete Handler Edit Code



3. Tương tác với khung View

❑ Sử dụng Slider Control

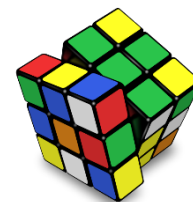
➤ Viết code cho hàm OnHScroll()

```
void ActionDlg::OnHScroll(UINT nSBCode, UINT nPos, CScrollBar* pScrollBar)
{
    // TODO: Add your message handler code here and/or call default
    CMFCmyAppView *pView = CMFCmyAppView::GetView();
    CSliderCtrl* pSlider = (CSliderCtrl*)pScrollBar;
    int position = pSlider->GetPos();
    float slider_step = 0.02;    // Bước của Slider là số nguyên. Do đó cần thêm 1 biến lưu giá trị bước
                                // thập phân mong muốn

    switch(pSlider->GetDlgCtrlID())
    {
    case IDC_SLIDER_X:
        pView->m_oglWindow.Obj1.SetPositionX(position*slider_step);
        break;
    case IDC_SLIDER_Y:
        pView->m_oglWindow.Obj1.SetPositionY(position*slider_step);
        break;
    }

    pView->m_oglWindow oglDrawScene();

    CDialogEx::OnHScroll(nSBCode, nPos, pScrollBar);
}
```



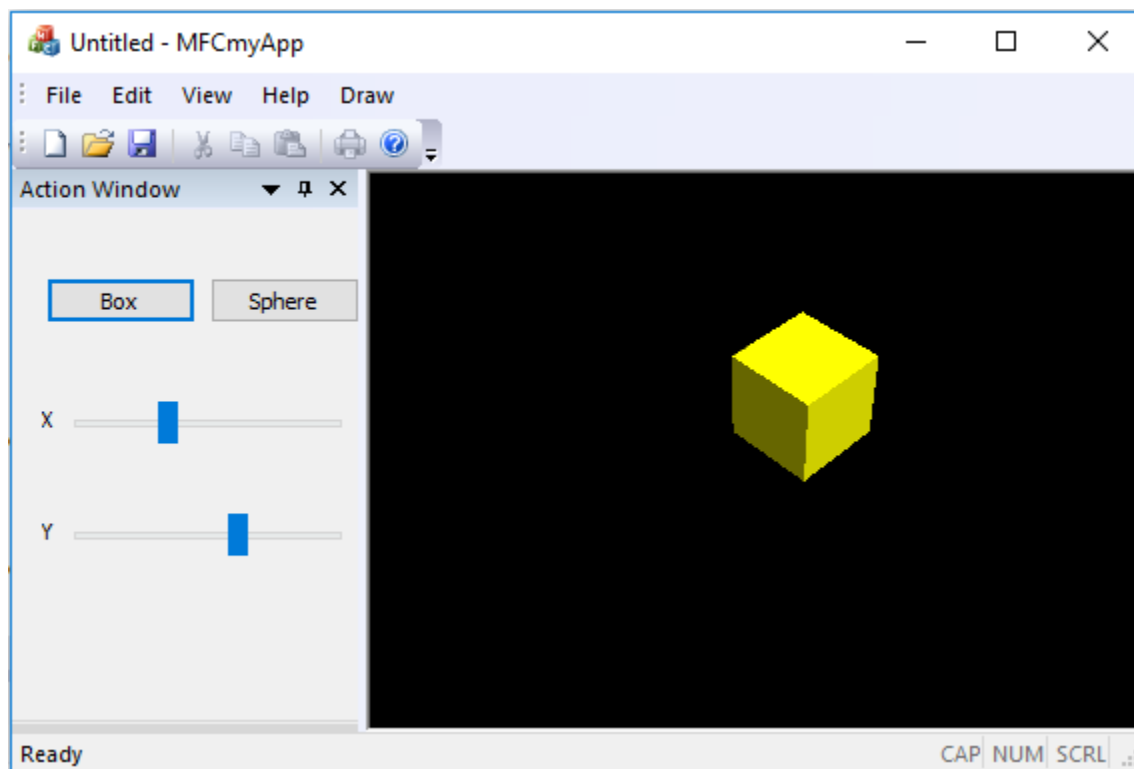
3. Tương tác với khung View

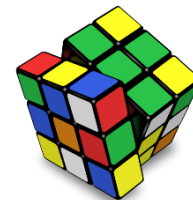
❑ Sử dụng Slider Control



Kết quả chạy:

- Khi kéo thanh trượt, khối hình dịch chuyển vị trí theo phương X hoặc Y

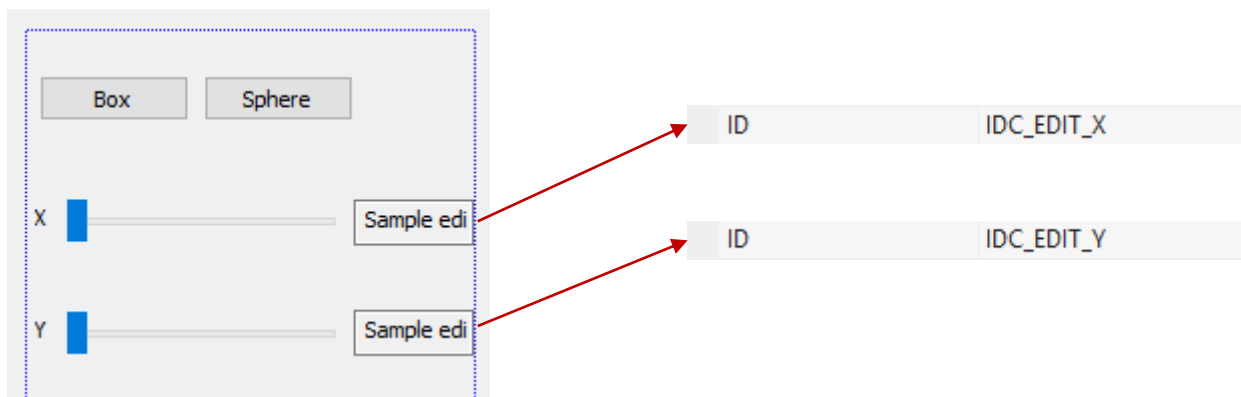




4. Tương tác từ View tới cửa sổ khác

❑ Thêm các hiển thị trên Dialog

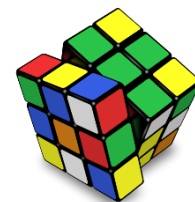
- Thêm các Edit Control vào Dialog như sau đây



- Thêm biến quản lý các Edit Control (Chuột phải -> Add Variable...)

```
CEdit EditX_ctrl;  
CEdit EditY_ctrl;
```

```
DDX_Control(pDX, IDC_EDIT_X, EditX_ctrl);  
DDX_Control(pDX, IDC_EDIT_Y, EditY_ctrl);
```

4. Tương tác từ View tới cửa sổ khác

❑ Thêm các hiển thị trên Dialog

- Khai báo thêm biến và sửa hàm OnHScroll() như sau

// File ...Dlg.h

```
float SliderStep;  
LPCTSTR NumberFormat;
```

// File ...Dlg.cpp

```
void ActionDlg::ActionDlg(CWnd* pParent /*=NULL*/)  
: CDialogEx(ActionDlg::IDD, pParent)  
{  
    SliderStep = 0.01;  
    NumberFormat = _T("%.2f");  
}
```

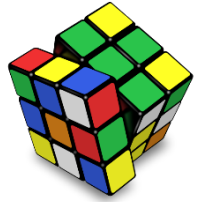
// File ...Dlg.cpp

```
void ActionDlg::OnHScroll(UINT nSBCode, UINT nPos, CScrollBar* pScrollBar)  
{  
    // TODO: Add your message handler code here and/or call default  
    CMFCmyAppView *pView = CMFCmyAppView::GetView();  
    CSliderCtrl* pSlider = (CSliderCtrl*)pScrollBar;  
    int position = pSlider->GetPos();  
    CString str;  
    str.Format(NumberFormat, position*SliderStep);  
  
    switch(pSlider->GetDlgCtrlID())  
    {  
        case IDC_SLIDER_X:  
            pView->m_oglWindow.Obj1.SetPositionX(position*SliderStep);  
            EditX_ctrl.SetWindowTextW(str);  
            break;  
        case IDC_SLIDER_Y:  
            pView->m_oglWindow.Obj1.SetPositionY(position*SliderStep);  
            EditY_ctrl.SetWindowTextW(str);  
            break;  
    }  
    pView->m_oglWindow.oglDrawScene();  
  
    CDialogEx::OnHScroll(nSBCode, nPos, pScrollBar);  
}
```



Kết quả chạy:

- Khi kéo thanh trượt, khối hình dịch chuyển và vị trí theo phương x, y hiển thị trong các ô Edit box



4. Tương tác từ View tới cửa sổ khác

❑ Cập nhật giá trị từ View ra Dialog

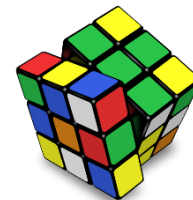
- Thêm hàm GetView() vào lớp ...Pane

// File ActionPane.h

```
public:  
    static ActionPane * GetView(UINT nID);
```

// File ActionPane.cpp

```
☐ ActionPane * ActionPane::GetView(UINT nID)  
{  
    CFrameWndEx * pFrame = (CFrameWndEx *) (AfxGetApp()->m_pMainWnd);  
  
    ActionPane * pPane = (ActionPane *) pFrame->GetPane(nID);  
  
    if ( !pPane )  
        return NULL;  
  
    if ( !pPane->IsKindOf( RUNTIME_CLASS(ActionPane) ) )  
        return NULL;  
  
    return pPane;  
}
```



4. Tương tác từ View tới cửa sổ khác

❑ Cập nhật giá trị từ View ra Dialog

➤ Thêm hàm OnInitialUpdate() cho lớp C...View

** Hàm OnInitialUpdate() của lớp CView có ý nghĩa tương tự hàm OnInitDialog() của lớp CDialog, có chức năng thực hiện các khởi tạo ban đầu*

The screenshot shows the Visual Studio Class Wizard and the Virtual Functions list. The Class Wizard is at the top, with the Project set to 'MFCMyApp' and the Class name set to 'CMFCMyAppView'. The Base class is 'CView', the Class declaration is 'mfcmyappview.h', and the Class implementation is 'mfcmyappview.cpp'. The Virtual Functions tab is selected, showing a list of virtual functions. The 'OnInitialUpdate' function is highlighted in the list. The 'Add Function' button is visible on the right.

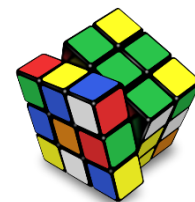
Class Wizard Fields:

- Project: MFCMyApp
- Class name: CMFCMyAppView
- Base class: CView
- Class declaration: mfcmyappview.h
- Class implementation: mfcmyappview.cpp

Virtual Functions List:

Virtual functions:	Overridden virtual functions:
OnDragLeave	AssertValid
OnDragOver	Dump
OnDraw	OnDraw
OnDrop	OnInitialUpdate
OnEndPrinting	PreCreateWindow
OnEndPrintPreview	
OnFinalRelease	
OnInitialUpdate	
OnNotify	
OnPrepareDC	

Buttons: Add Function, Delete Function, Edit Code



4. Tương tác từ View tới cửa sổ khác

❑ Cập nhật giá trị từ View ra Dialog

➤ Viết code cho hàm OnInitialUpdate()

```
#include "ActionPane.h"
...

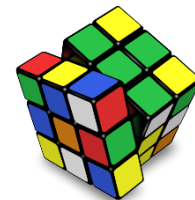
void CMFCmyAppView::OnInitialUpdate()
{
    CView::OnInitialUpdate();

    // TODO: Add your specialized code here and/or call the base class
    ActionPane *pActionPane = ActionPane::GetView(ID_VIEW_ACTIONWINDOW); //-> Phải là ID đã dùng khi khởi tạo Pane trong
                                                                    hàm OnCreate() của lớp CMainFrame

    int SliderXpos = m_oglWindow.Obj1.GetX()/pActionPane->m_ActDlg.SliderStep;
    int SliderYpos = m_oglWindow.Obj1.GetY()/pActionPane->m_ActDlg.SliderStep;

    pActionPane->m_ActDlg.SliderX_ctrl.SetRange(-100, 100, TRUE); // Thiết lập giới hạn Min, Max của Slider
    pActionPane->m_ActDlg.SliderX_ctrl.SetPos(SliderXpos);         // Thiết lập vị trí con trượt của Slider
    pActionPane->m_ActDlg.SliderY_ctrl.SetRange(-100, 100, TRUE);
    pActionPane->m_ActDlg.SliderY_ctrl.SetPos(SliderYpos);

    CString str;
    str.Format(pActionPane->m_ActDlg.NumberFormat, m_oglWindow.Obj1.GetX());
    pActionPane->m_ActDlg.EditX_ctrl.SetWindowTextW(str); // Hiển thị ra Editbox
    str.Format(pActionPane->m_ActDlg.NumberFormat, m_oglWindow.Obj1.GetY());
    pActionPane->m_ActDlg.EditY_ctrl.SetWindowTextW(str);
}
```



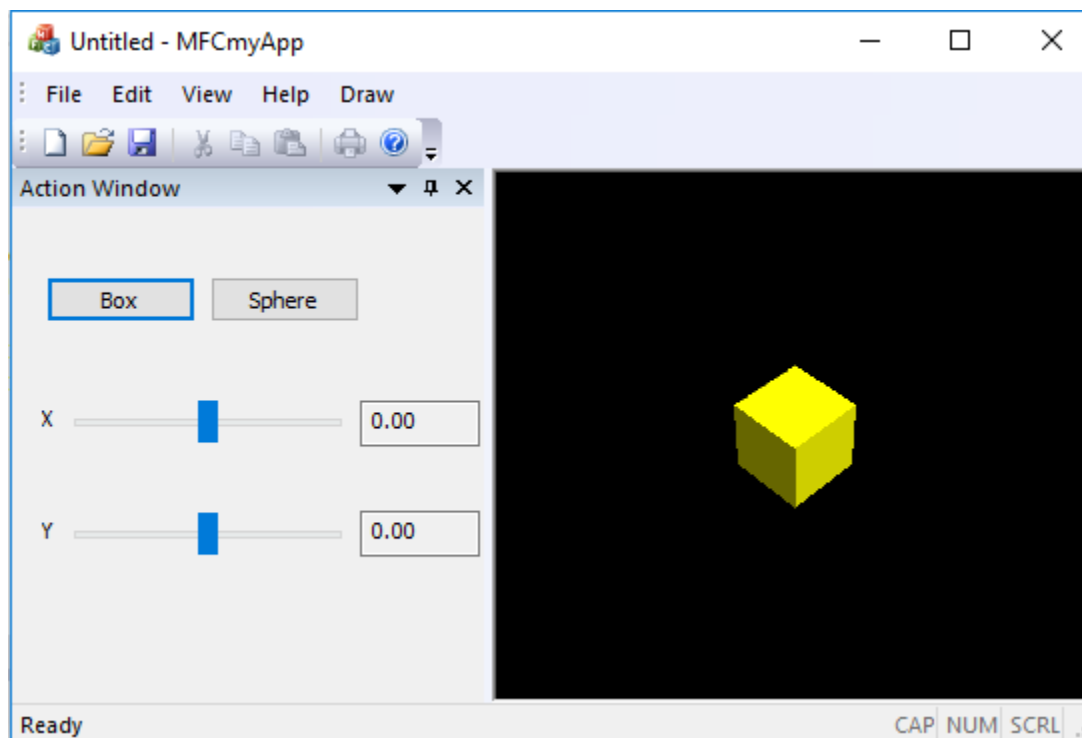
4. Tương tác từ View tới cửa sổ khác

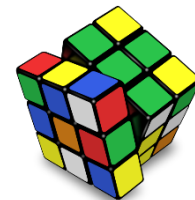
❑ Cập nhật giá trị từ View ra Dialog



Kết quả chạy:

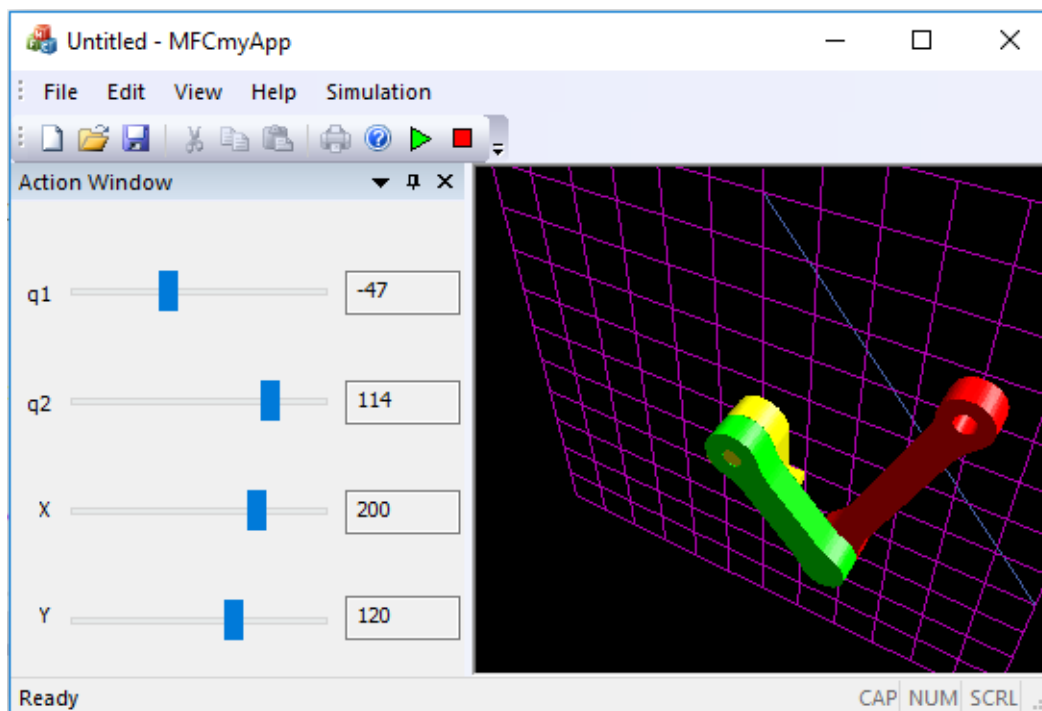
- Vị trí theo phương x, y của hình khối được cập nhật ra Slider và Edit box ngay từ đầu





Tổng hợp kiến thức

- ❑ Áp dụng các nội dung đã trình bày, xây dựng ứng dụng MFC kiểu SDI, mô phỏng robot 2 bậc tự do, có ô cửa điều khiển với các thanh trượt, khi di chuyển các con trượt:
 - A. Các góc khớp của Robot thay đổi từ -180° đến 180°
 - B. Điểm tác động cuối của Robot di chuyển theo phương X, phương Y





Tổng hợp kiến thức

- ❑ Áp dụng các nội dung đã trình bày, xây dựng ứng dụng MFC kiểu SDI, mô phỏng robot 2 bậc tự do, có ô cửa điều khiển với các thanh trượt...
- Gợi ý: khi kéo điểm tác động cuối của robot trượt theo phương X hoặc phương Y, có những vị trí mà bài toán động học ngược vô nghiệm, mô hình robot không vẽ được. Để tránh trường hợp này, có thể sửa đổi một số hàm của lớp robot.

```
bool Robot::InvKin(double par_x, double par_y)
{
    double cosq2 = (par_x*par_x+par_y*par_y-l1*l1-l2*l2)/(2*l1*l2);
    if (cosq2>1 || cosq2<-1)
        return FALSE;
    x = par_x;
    y = par_y;
    q2=acos(cosq2);
    q1=atan2(y,x)-atan2(l2*sin(q2),l1+l2*cos(q2));
    return TRUE;
}

bool Robot::Setpos(double par1, double par2)
{
    return InvKin(par1,par2);
}

bool Robot::Setx(double par_x)
{
    return InvKin(par_x, y);
}

bool Robot::Sety(double par_y)
{
    return InvKin(x, par_y);
}
```

Hết Bài 10

