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

Bài 9:
Cài đặt OpenGL cho ứng dụng kiểu SDI của MFC

PHẠM MINH QUÂN

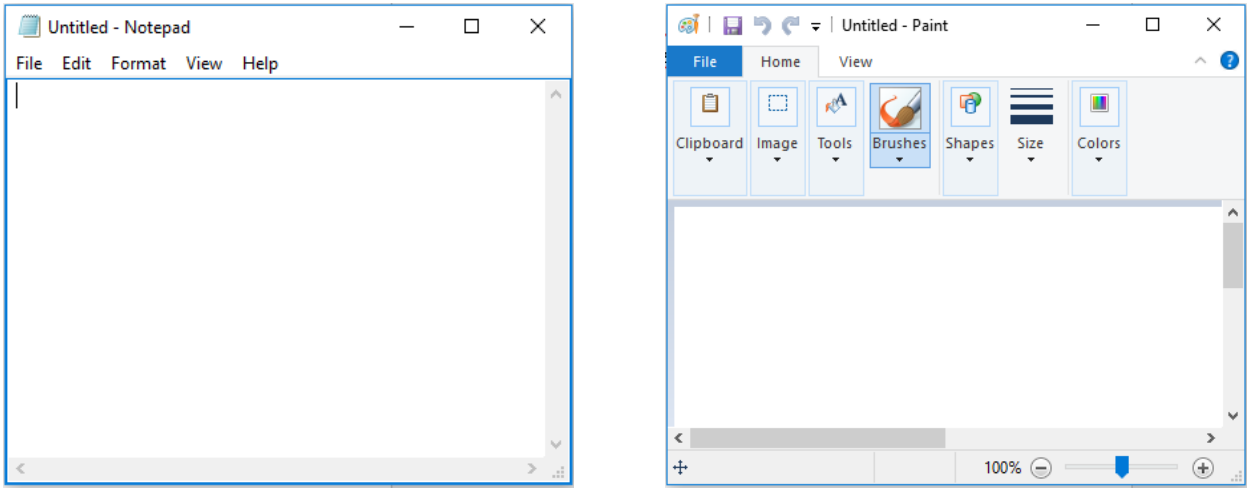
mquan.ph@gmail.com

Nội dung

1. Kiểu ứng dụng SDI của MFC
2. Cài đặt OpenGL cho ứng dụng kiểu SDI
3. Thiết kế menu, toolbar
4. Mở rộng
   4.1. Triển khai ứng dụng (đã hoàn thành) sang các máy tính (sử dụng Windows) khác.
   4.2. Một số phần mềm mô phỏng robot

1. Kiểu ứng dụng SDI của MFC

- Giới thiệu giao diện SDI (Single Document Interface)
  - Mỗi khung cửa sổ chỉ mở một tài liệu, không có cửa sổ con.
  - Cho phép thay đổi kích thước cửa sổ
  - Ví dụ: Notepad, Paint…

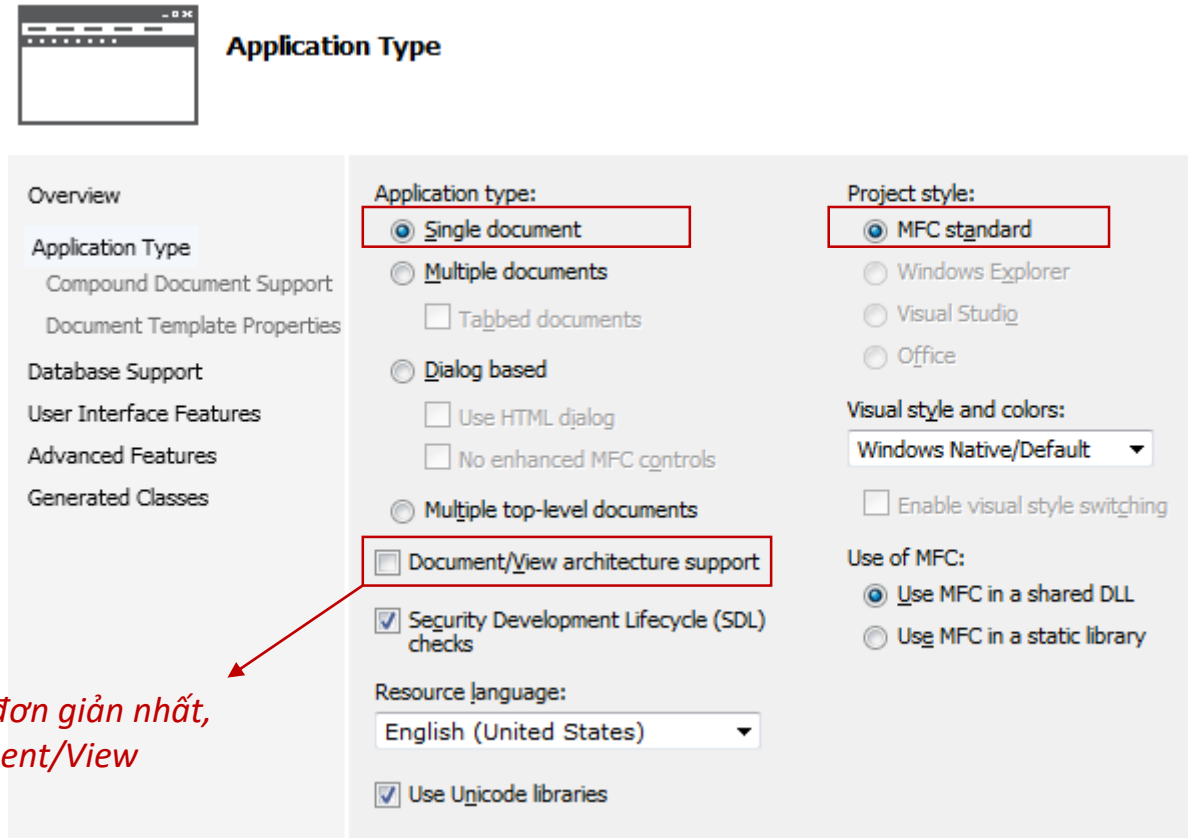


Các ví dụ về phần mềm có kiểu giao diện SDI

1. Kiểu ứng dụng SDI của MFC

❑Tạo ứng dụng MFC kiểu SDI

➢ Tạo project mới MFC Application. Chọn kiểu ứng dụng là Single document.

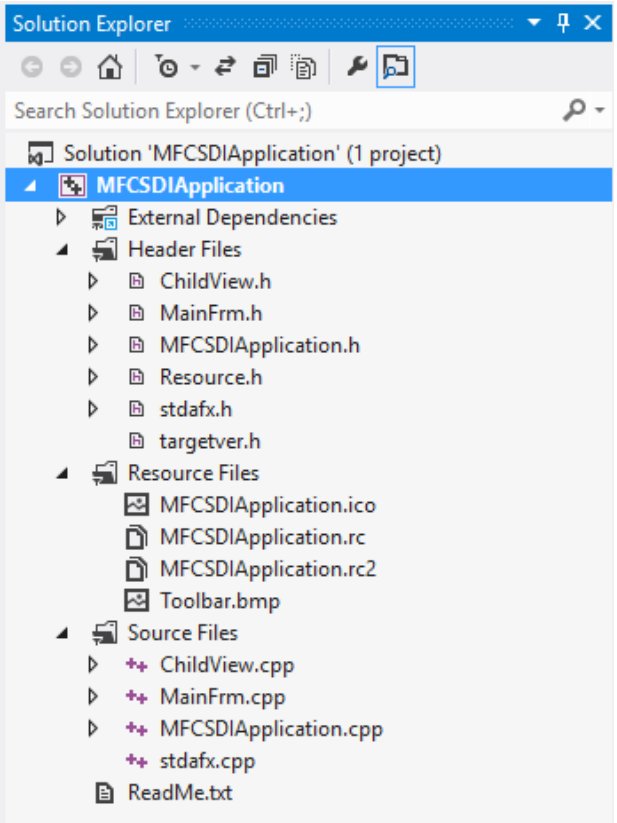


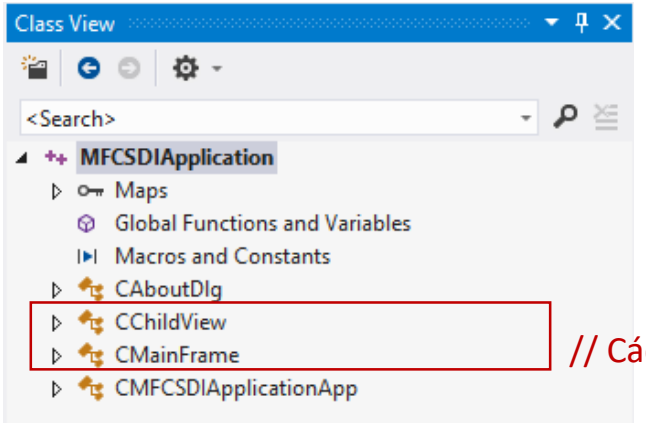
Để tạo ứng dụng đơn giản nhất, ta bỏ chọn Document/View

1. Kiểu ứng dụng SDI của MFC

❑Tạo ứng dụng MFC kiểu SDI

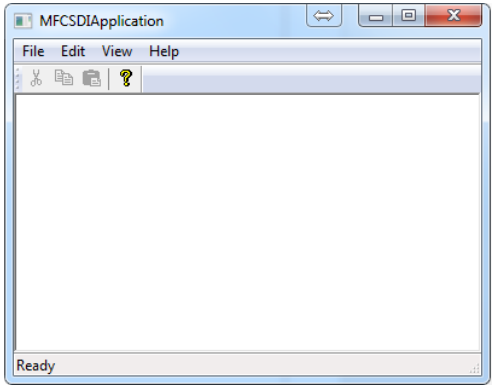
➢ Quan sát các file, các lớp được tạo sẵn.





// Các lớp cần lưu ý

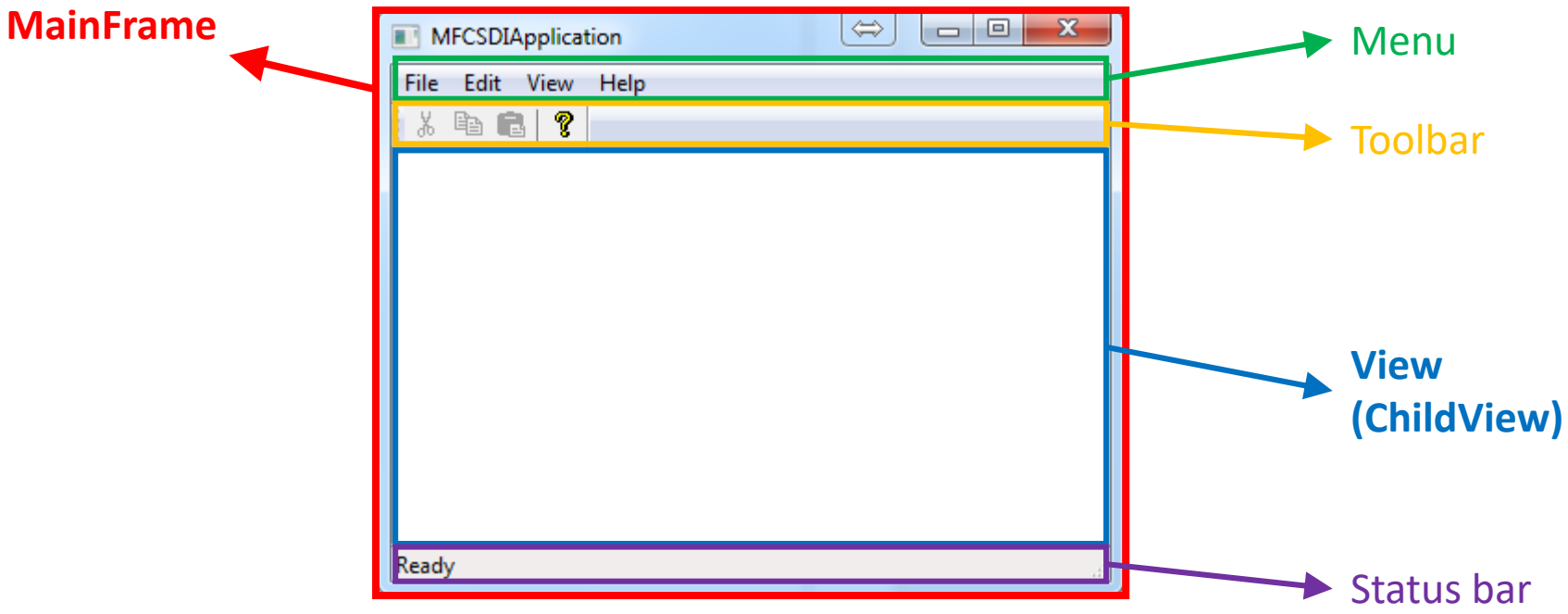
Kết quả chạy:



1. Kiểu ứng dụng SDI của MFC

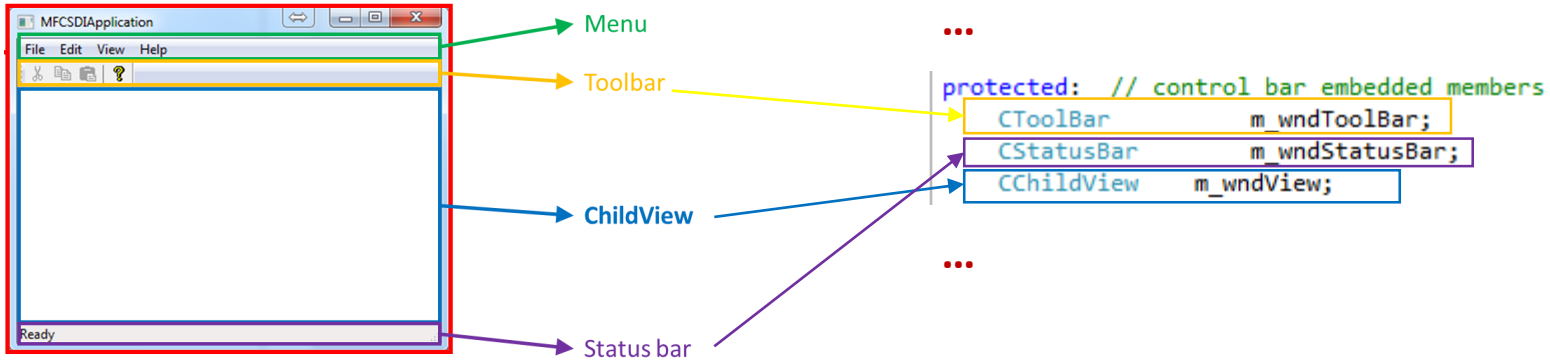
❑Tạo ứng dụng MFC kiểu SDI

- ➢ Quan sát kết quả chạy.



1. Kiểu ứng dụng SDI của MFC

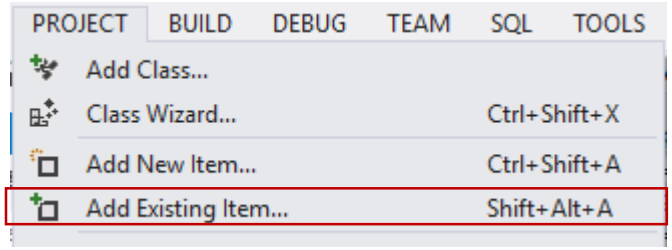
- Tạo ứng dụng MFC kiểu SDI
  - Quan sát lớp CMainFrame.

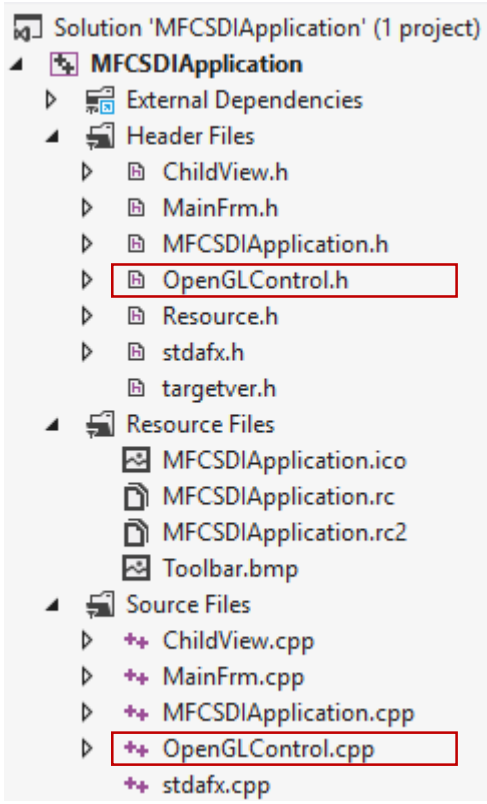


2. Cài đặt OpenGL cho ứng dụng kiểu SDI

❑Thêm lớp quản lý OpenGL

- Copy file “OpenGLControl.h” và “OpenGLControl.cpp” đã xây dựng trong Bài 6 vào thư mục của Project và dùng lệnh Add Existing Items… để thêm vào project.





2. Cài đặt OpenGL cho ứng dụng kiểu SDI

❑Cài đặt OpenGL vào ChildView

➢ Khai báo đối tượng của lớp COpenGLControl làm thuộc tính của lớp CChildView

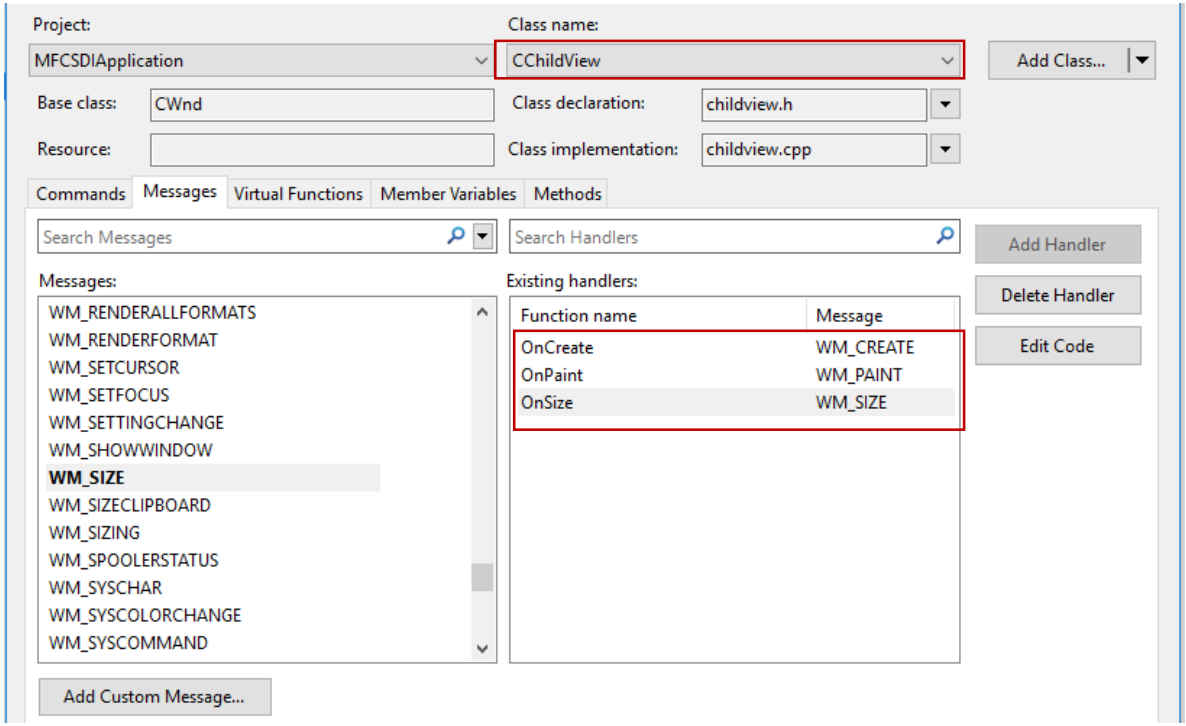
```
// File “ChildView.h”

#include "OpenGLControl.h"
...
    COpenGLControl m_oglWindow;
...
```

2. Cài đặt OpenGL cho ứng dụng kiểu SDI

❑Cài đặt OpenGL vào ChildView

- Để cài đặt OpenGL, ta cần sửa code của 3 hàm afx_msg của lớp CChildView là: OnCreate(), OnSize(), OnPaint()

  (Trong đó, hàm OnPaint() là có sẵn còn các hàm OnCreate() và OnSize() cần được thêm vào bằng Class Wizard)



2. Cài đặt OpenGL cho ứng dụng kiểu SDI

❑Cài đặt OpenGL vào ChildView

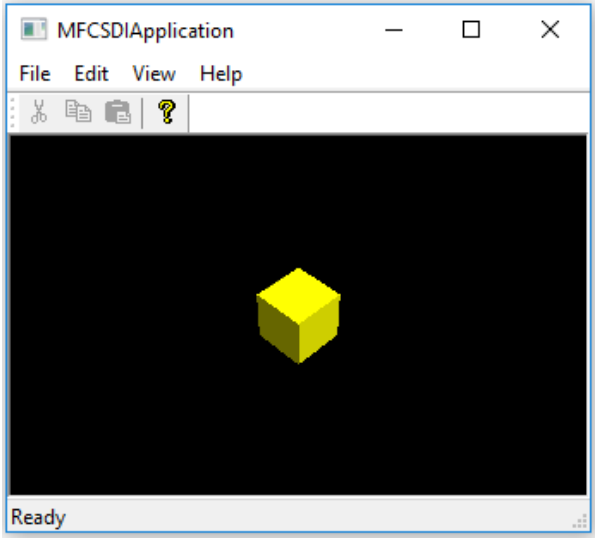
➢ Sửa code của các hàm OnCreate(), OnSize(), OnPaint() của lớp CChildView như sau:

```
void CChildView::OnPaint()
{
    CPaintDC dc(this); // device context for painting

    // TODO: Add your message handler code here
    m_oglWindow.oglDrawScene();
}

int CChildView::OnCreate(LPCREATESTRUCT lpCreateStruct)
{
    if (CWnd::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 CChildView::OnSize(UINT nType, int cx, int cy)
{
    CWnd::OnSize(nType, cx, cy);

    // TODO: Add your message handler code here
    m_oglWindow.MoveWindow(0, 0, cx, cy);
}
```

Kết quả chạy:

- Đồ họa OpenGL hiện trong khung View

- Khi thay đổi kích thước cửa sổ thì hình đồ họa điều chỉnh kích thước theo



3. Thiết kế menu, toolbar

❖Ví dụ: *Tạo các lệnh trên menu và toolbar thực hiện vẽ các hình khối khác nhau trong khung đồ họa OpenGL: hình hộp, hình cầu…*

❑ Xây dựng một lớp mới để quản lý các hình khối: lớp OglObject

//File OglObject.h

```
#pragma once

#define BOX     1
#define SPHERE  2

class OglObject
{
public:
    OglObject(void);
    ~OglObject(void);
protected:
    int type;
public:
    void SetType(int);
    void Draw();
};
```

//File OglObject.cpp

```
#include "stdafx.h"
#include "OglObject.h"
#include "glut.h"

OglObject::OglObject(void) {type = BOX;}

OglObject::~OglObject(void) {}

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

void OglObject::Draw()
{
    switch(type)
    {
    case BOX:
        glutSolidCube (1.0);
        break;
    case SPHERE:
        glutSolidSphere(1.0, 30, 30);
        break;
    }
}
```

3. Thiết kế menu, toolbar

❑ Sử dụng đối tượng của lớp OglObject để vẽ hình khối

//File OpenGLControl.h

```
#include "OglObject.h"

...

    OglObject Obj1;

...
```

//File OpenGLControl.cpp

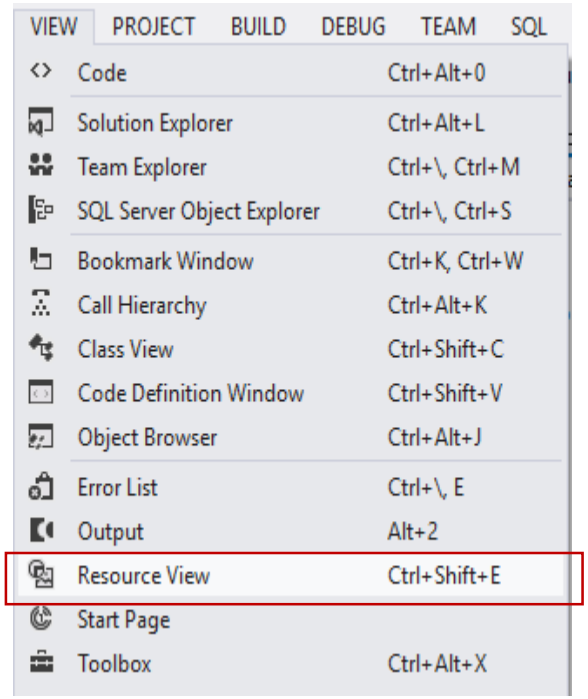
```
void COpenGLControl::oglDrawScene(void)
{
    // Clear color and depth buffer bits
    glClear(GL_COLOR_BUFFER_BIT | GL_DEPTH_BUFFER_BIT);

    GLfloat mat_color[] = { 1.0, 1.0, 0.0 };
    GLfloat mat_specular[] = { 1.0, 1.0, 1.0 };

    glMaterialfv(GL_FRONT, GL_AMBIENT_AND_DIFFUSE, mat_color);
    glMaterialfv(GL_FRONT, GL_SPECULAR, mat_specular);
    glMaterialf(GL_FRONT, GL_SHININESS, 30);

    Obj1.Draw();

    // Swap buffers
    SwapBuffers(hdc);
}
```

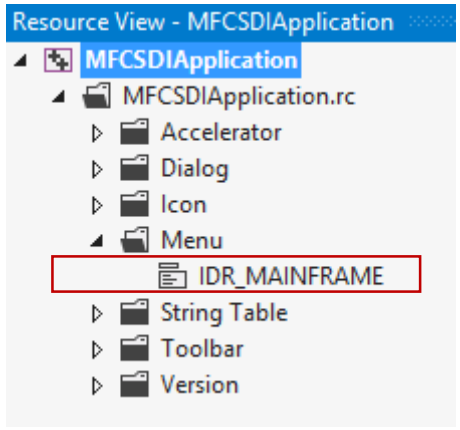
Kết quả chạy: Như lần chạy trước

3. Thiết kế menu, toolbar

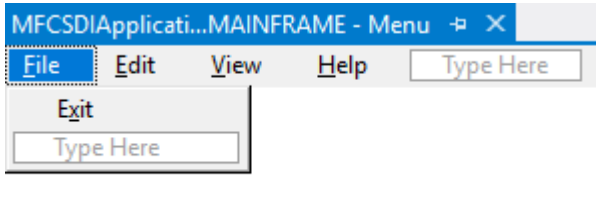
❑Thiết kế Menu

➢ Mở cửa sổ Resource View -> Click đúp để mở Menu/IDR_MAINFRAME





Cửa sổ Resource View

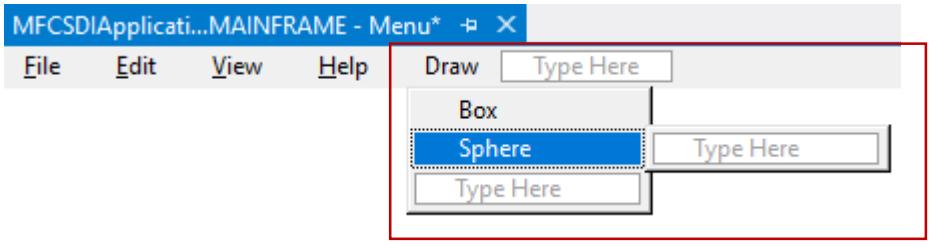


Giao diện thiết kế Menu IDR_MAINFRAME

3. Thiết kế menu, toolbar

❑Thiết kế Menu

- Thêm các lệnh vẽ khối hộp, khối cầu vào menu:



- Thiết lập thông số (properties) cho các nút lệnh mới trong menu:

Appearance	
Caption	Box
Checked	False
Enabled	True
Grayed	False
Popup	False
Behavior	
Break	None
Right Justify	False
Right Order	False
Misc	
(Name)	Menu Editor
Help	False
ID	ID_DRAW_BOX

Properties của nút lệnh vẽ Hộp

Appearance	
Caption	Sphere
Checked	False
Enabled	True
Grayed	False
Popup	False
Behavior	
Break	None
Right Justify	False
Right Order	False
Misc	
(Name)	Menu Editor
Help	False
ID	ID_DRAW_SPHERE

Properties của nút lệnh vẽ Cầu

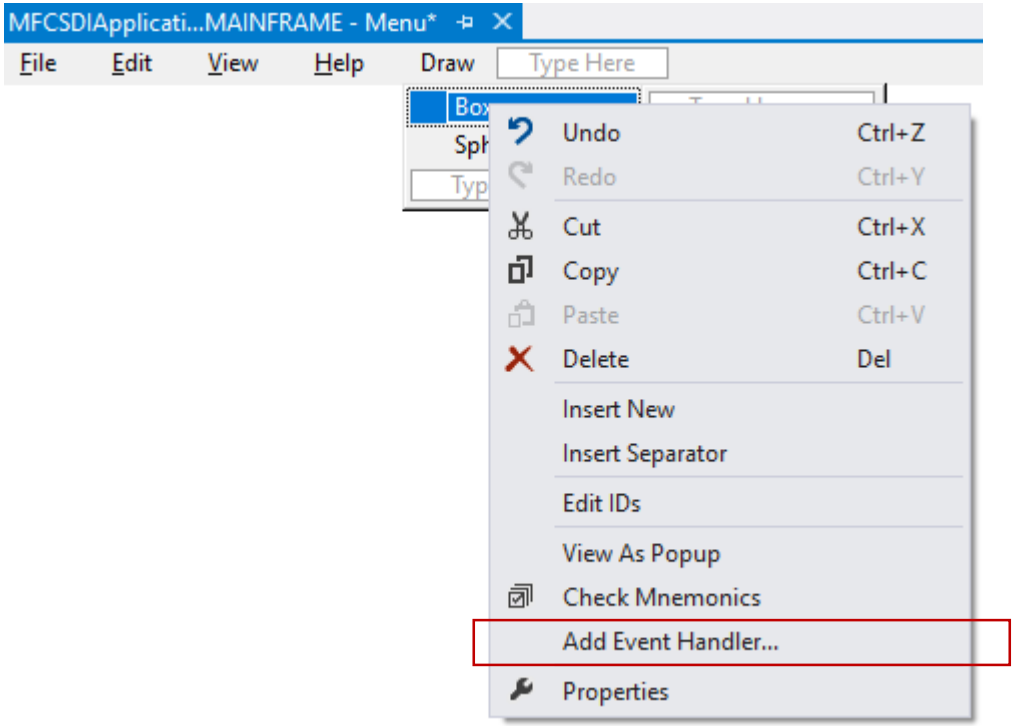
// ID là thông tin quan trọng nhất

3. Thiết kế menu, toolbar

❑Thiết kế Menu

➢ Tạo hàm thực thi các nút lệnh mới trong menu:

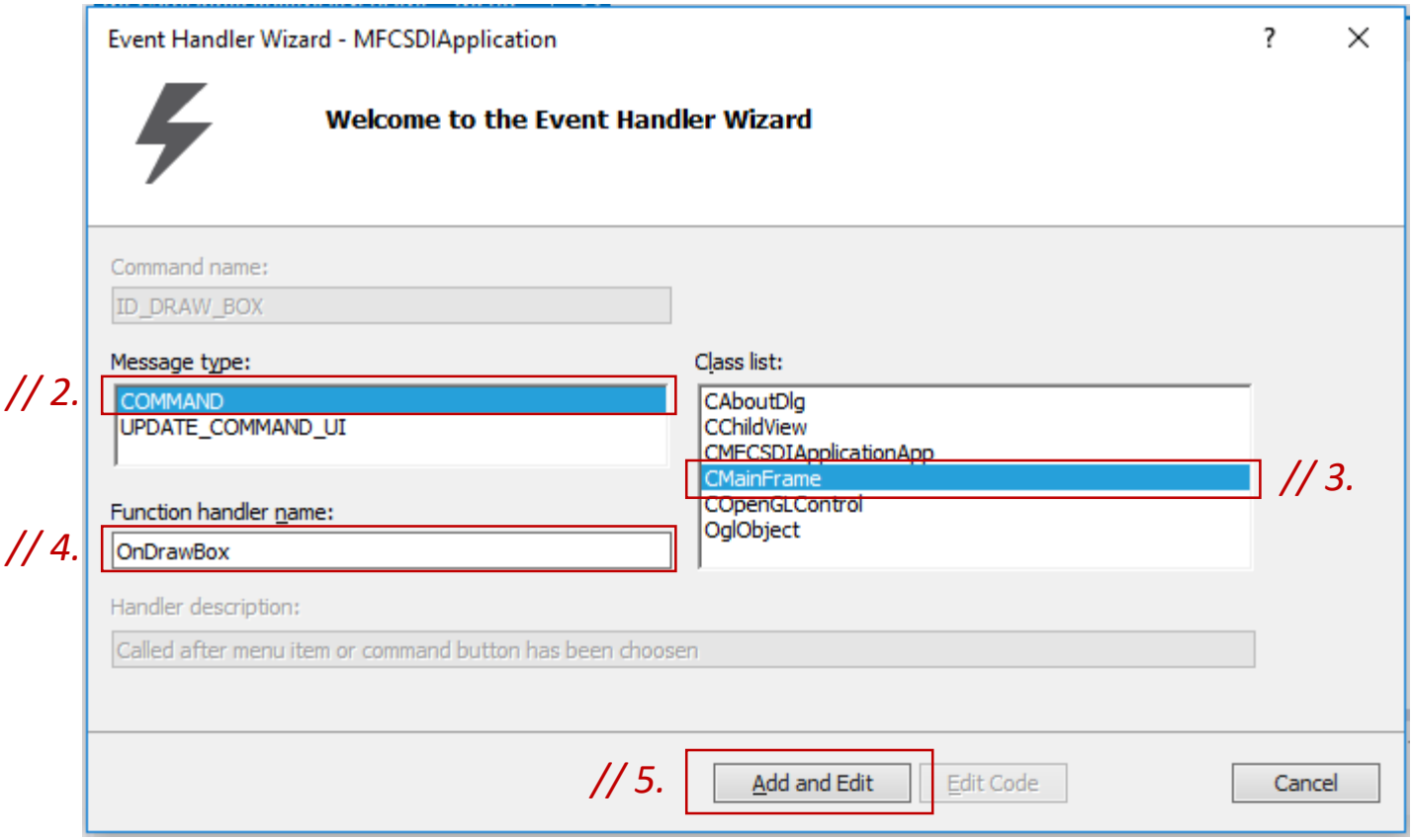
// 1. Click chuột phải vào nút lệnh -> Add Event Handler…



3. Thiết kế menu, toolbar

❑Thiết kế Menu

➢ Tạo hàm thực thi các nút lệnh mới trong menu:



3. Thiết kế menu, toolbar

❑Thiết kế Menu

➢ Tạo hàm thực thi các nút lệnh mới trong menu:

// 6. Quan sát các dòng lệnh mà chương trình tạo thêm:
(ta có thể tự khai báo bằng cách viết code thủ công các dòng lệnh này)

File MainFrm.h

```
public:
    afx_msg void OnDrawBox();
    afx_msg void OnDrawSphere();
```

File MainFrm.cpp

```
BEGIN_MESSAGE_MAP(CMainFrame, CFrameWnd)
...
    ON_COMMAND(ID_DRAW_BOX, &CMainFrame::OnDrawBox)
    ON_COMMAND(ID_DRAW_SPHERE, &CMainFrame::OnDrawSphere)
END_MESSAGE_MAP()

...

void CMainFrame::OnDrawBox()
{
    // TODO: Add your command handler code here
}

void CMainFrame::OnDrawSphere()
{
    // TODO: Add your command handler code here
}
```

3. Thiết kế menu, toolbar

❑Thiết kế Menu

➢ Viết code cho hàm thực thi các nút lệnh mới:

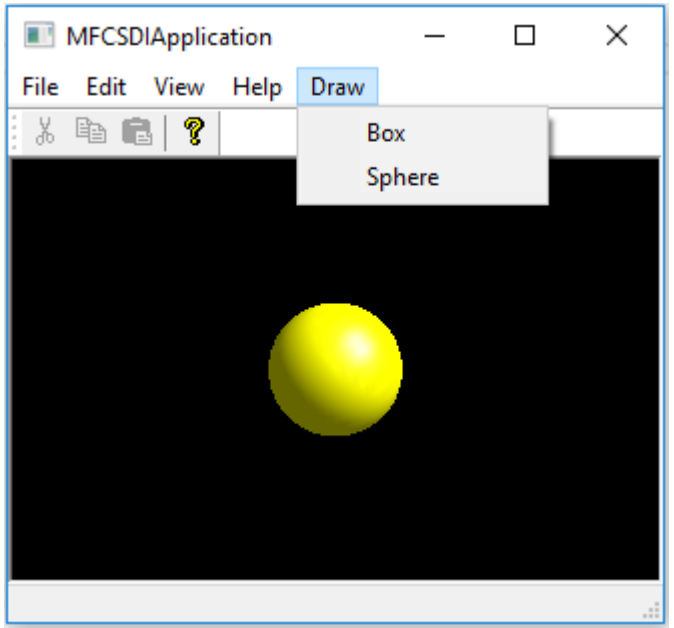
File MainFrm.cpp

```
void CMainFrame::OnDrawBox()
{
    // TODO: Add your command handler code here
    m_wndView.m_oglWindow.Obj1.SetType(BOX);
    m_wndView.m_oglWindow.oglDrawScene();
}

void CMainFrame::OnDrawSphere()
{
    // TODO: Add your command handler code here
    m_wndView.m_oglWindow.Obj1.SetType(SPHERE);
    m_wndView.m_oglWindow.oglDrawScene();
}
```

Kết quả chạy:

- Xuất hiện lệnh mới trong Menu

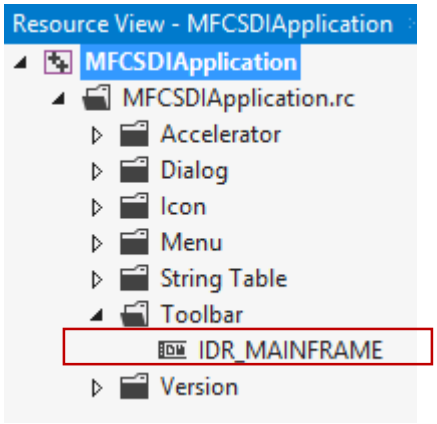
- Khi click vào lệnh tương ứng, hình khối trong khung đồ họa OpenGL thay đổi

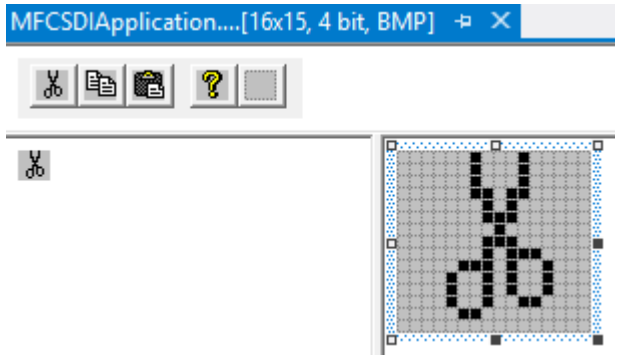


3. Thiết kế menu, toolbar

❑Thiết kế Toolbar

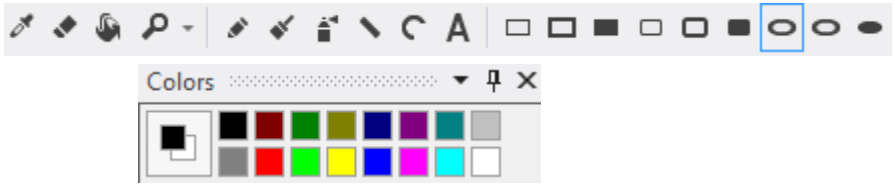
- Mở cửa sổ Resource View -> Click đúp để mở Toolbar/IDR_MAINFRAME

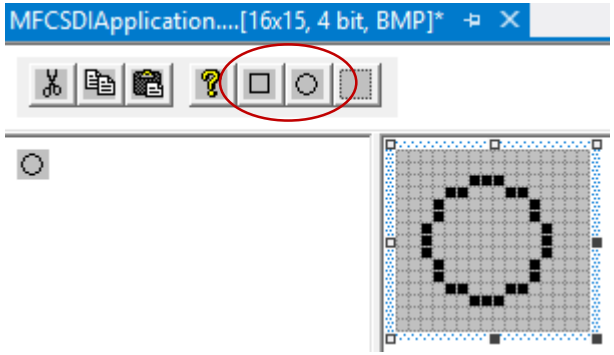




- Thêm các nút vẽ khối hộp, khối cầu trên toolbar

(sử dụng công cụ do Visual Studio cung cấp để vẽ icon của nút)

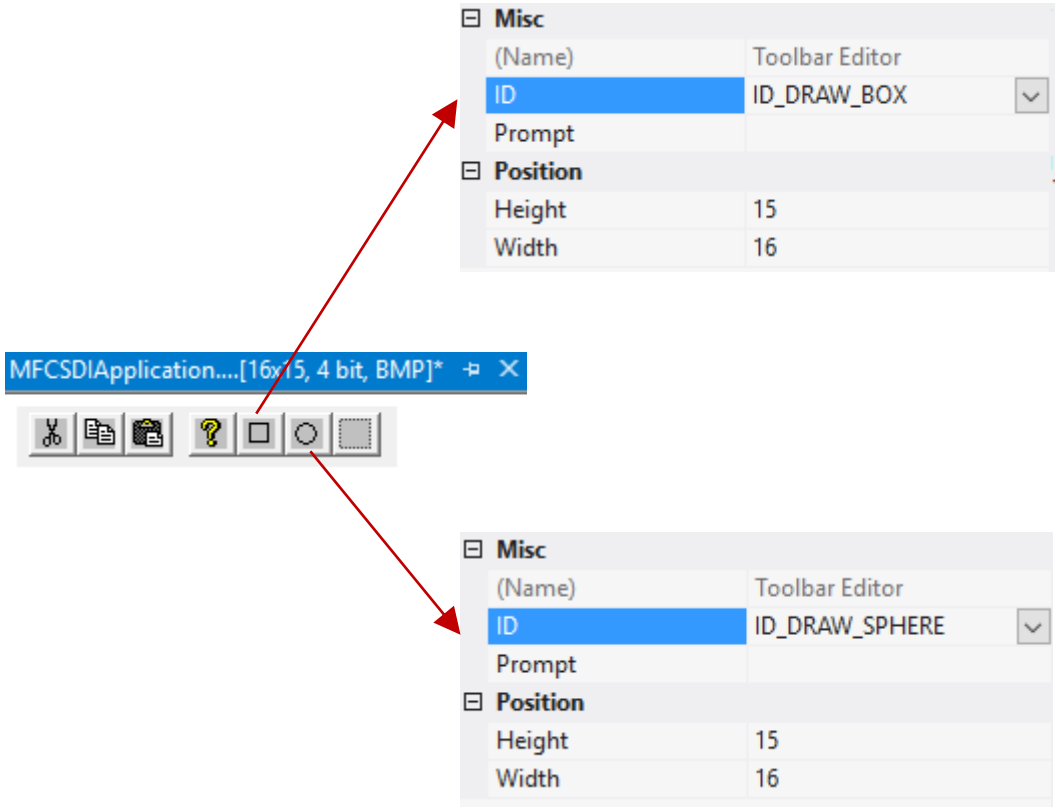




3. Thiết kế menu, toolbar

❑Thiết kế Toolbar

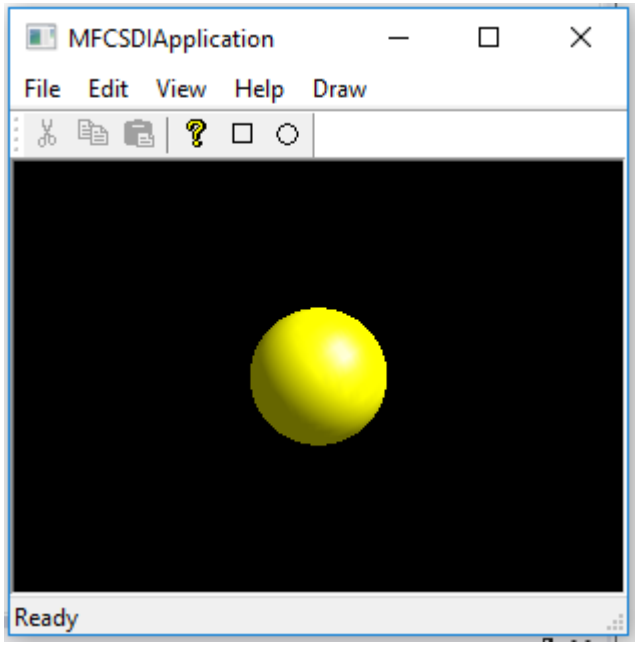
➢ Thiết lập ID của các nút trên Toolbar trùng với ID của các lệnh trên Menu



Kết quả chạy:

- Xuất hiện nút mới trong Toolbar

- Khi click vào nút tương ứng, hình khối trong khung đồ họa OpenGL thay đổi

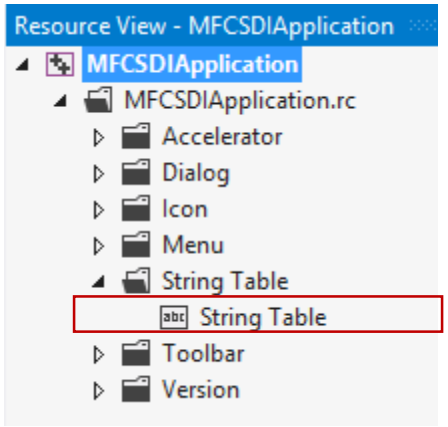


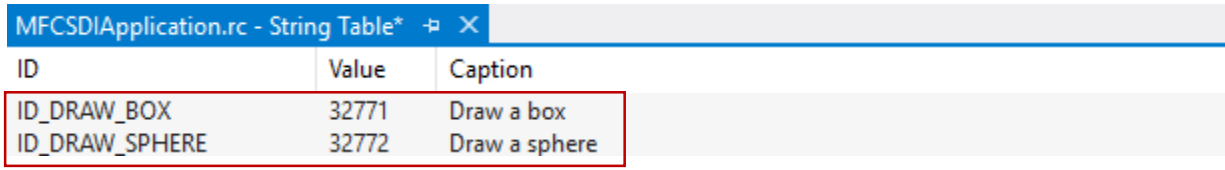
3. Thiết kế menu, toolbar

❑Thiết kế Toolbar

➢ Thêm chú giải cho các nút lệnh:

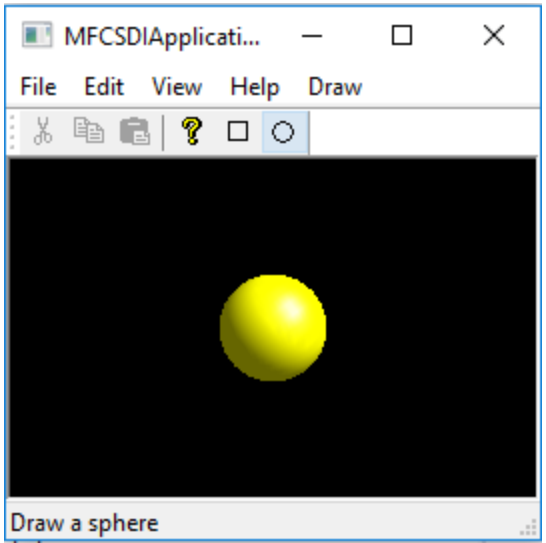
Mở cửa sổ Resource View -> Click đúp để mở String Table/String Table -> Chọn ID của các nút lệnh và thêm Caption mong muốn





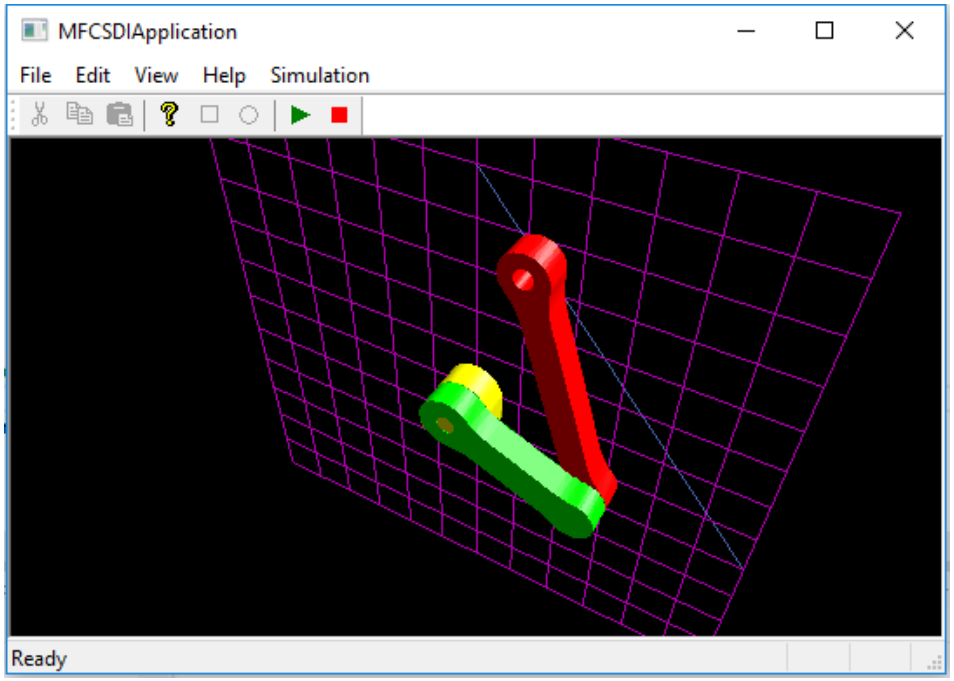
Kết quả chạy:

- Khi đưa chuột qua các nút lệnh, xuất hiện dòng chú thích ở Status bar



Tổng hợp kiến thức

- Áp dụng các nội dung trình bày trong Bài 9 và Bài 7, xây dựng được ứng dụng MFC kiểu SDI, mô phỏng robot 2 bậc tự do, có các nút lệnh trên toolbar và menu cho phép chạy và dừng mô phỏng:



4. Mở rộng

4.1. Triển khai ứng dụng (đã hoàn thành) sang các máy tính (sử dụng Windows) khác

- Có 2 phương pháp: Tạo file setup và Triển khai trực tiếp
  - Tạo file setup: (xem hướng dẫn ở link sau)

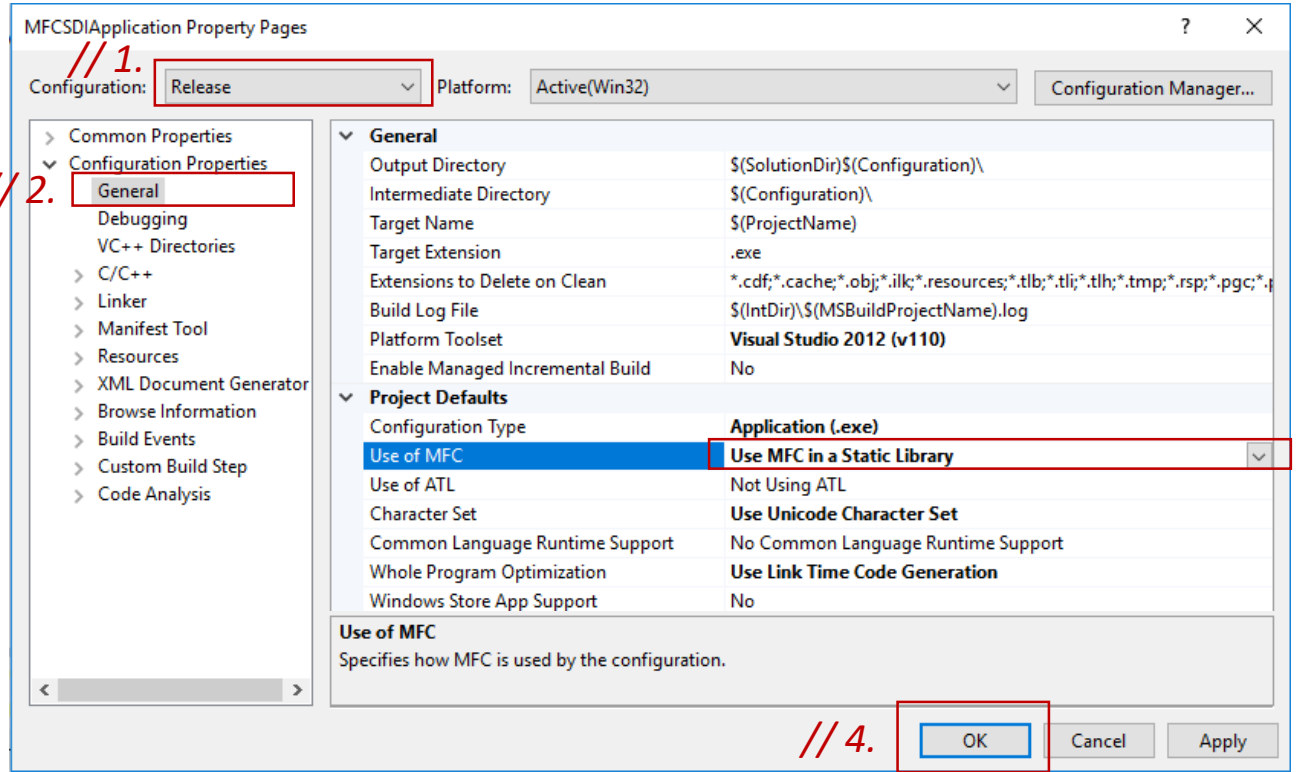
    https://msdn.microsoft.com/en-us/library/dd293568.aspx
  - Triển khai trực tiếp: (xem hướng dẫn ở link sau)

    https://msdn.microsoft.com/en-us/library/dd293565.aspx

4. Mở rộng

4.1. Triển khai ứng dụng (đã hoàn thành) sang các máy tính (sử dụng Windows) khác

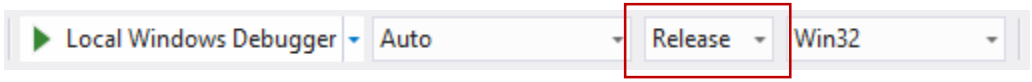
- Ví dụ cách Triển khai trực tiếp: (Với 1 ứng dụng MFC có sử dụng thư viện OpenGL)
  - Mở Project Property và thay đổi thông tin như sau:



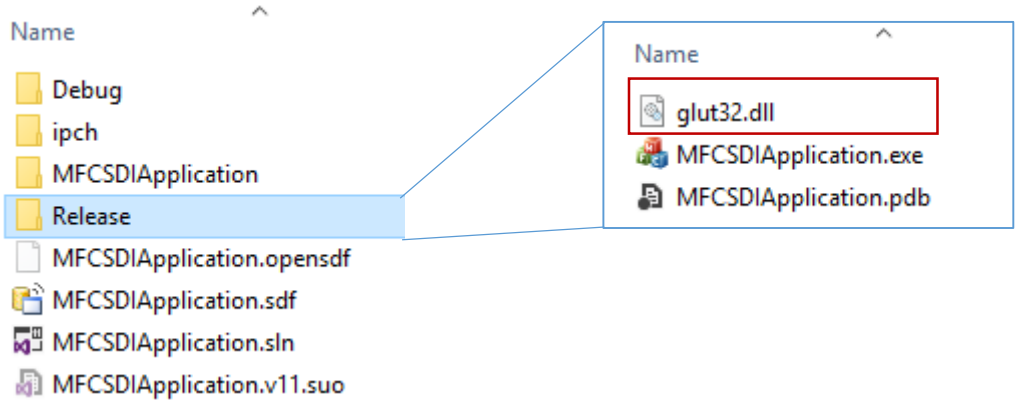
4. Mở rộng

4.1. Triển khai ứng dụng (đã hoàn thành) sang các máy tính (sử dụng Windows) khác

- Ví dụ cách Triển khai trực tiếp: (Với 1 ứng dụng MFC có sử dụng thư viện OpenGL)
  - Chọn chế độ Build là Release và Build lại project



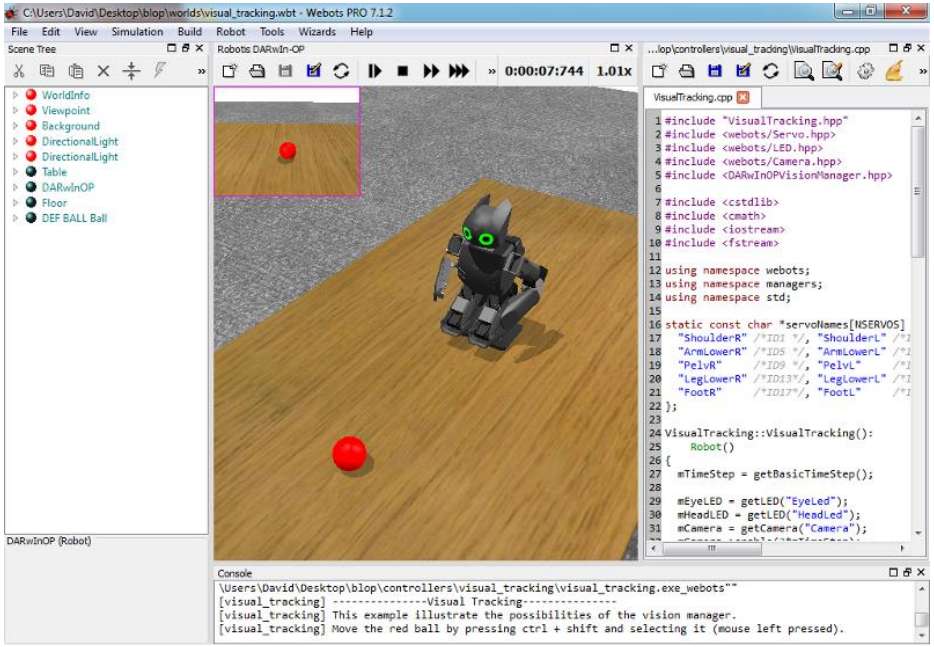
  - Copy file .dll của thư viện dùng thêm (OpenGL) vào thư mục Release của project



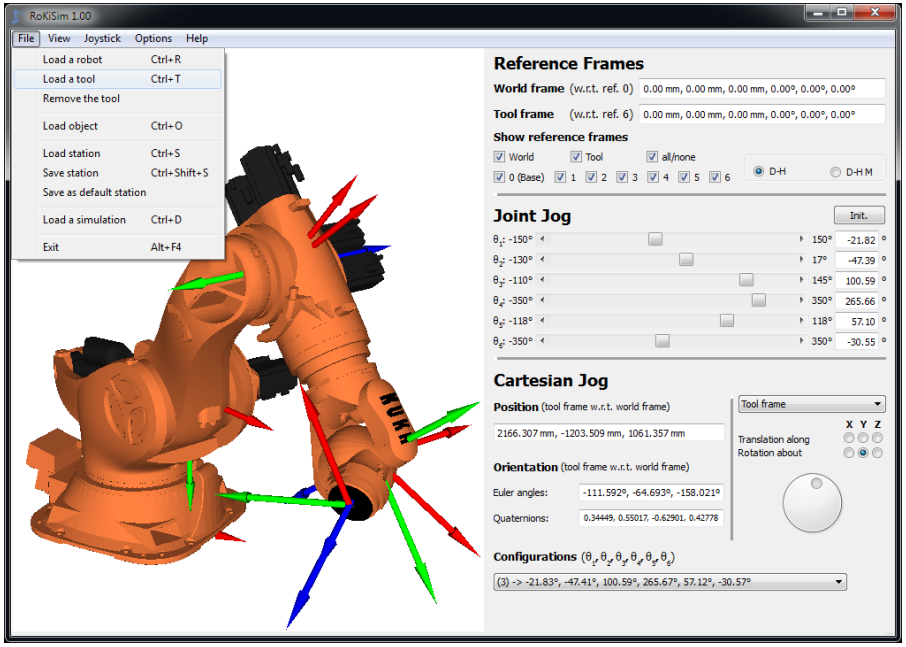
  - Copy thư mục Release sang bất cứ máy tính Windows nào khác.
    Chạy chương trình bằng cách click đúp file .exe

4. Mở rộng

4.2. Một số phần mềm mô phỏng Robot



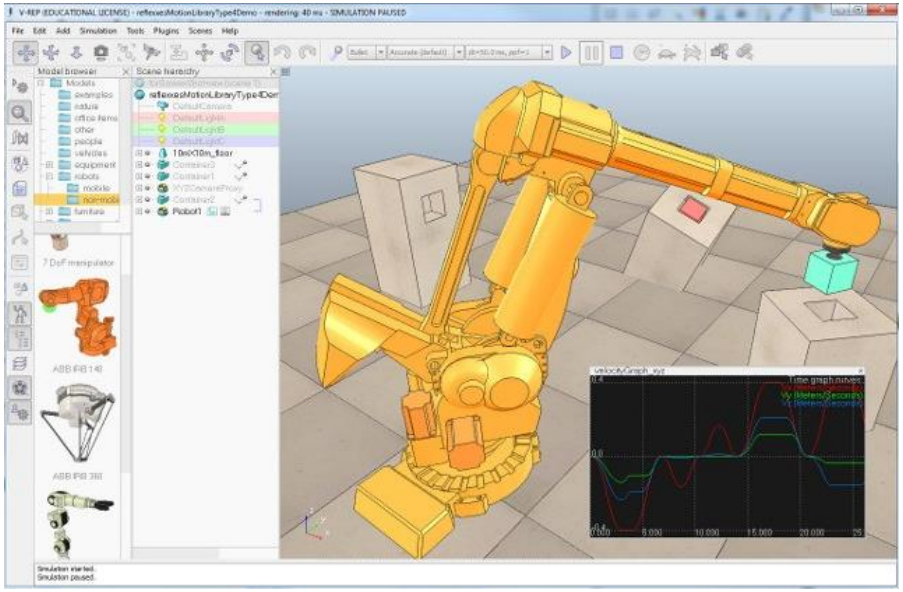
Webots



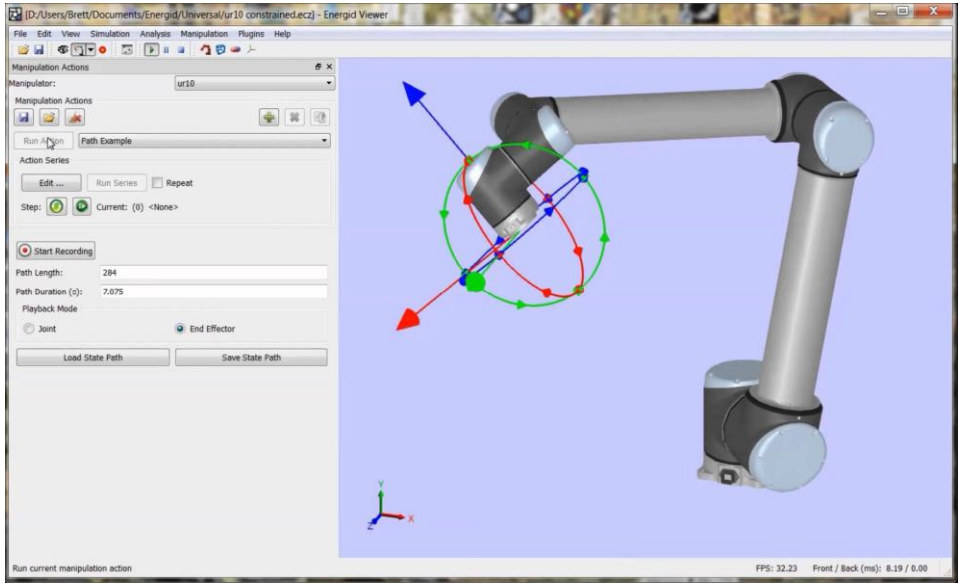
RoKiSim

4. Mở rộng

4.2. Một số phần mềm mô phỏng Robot



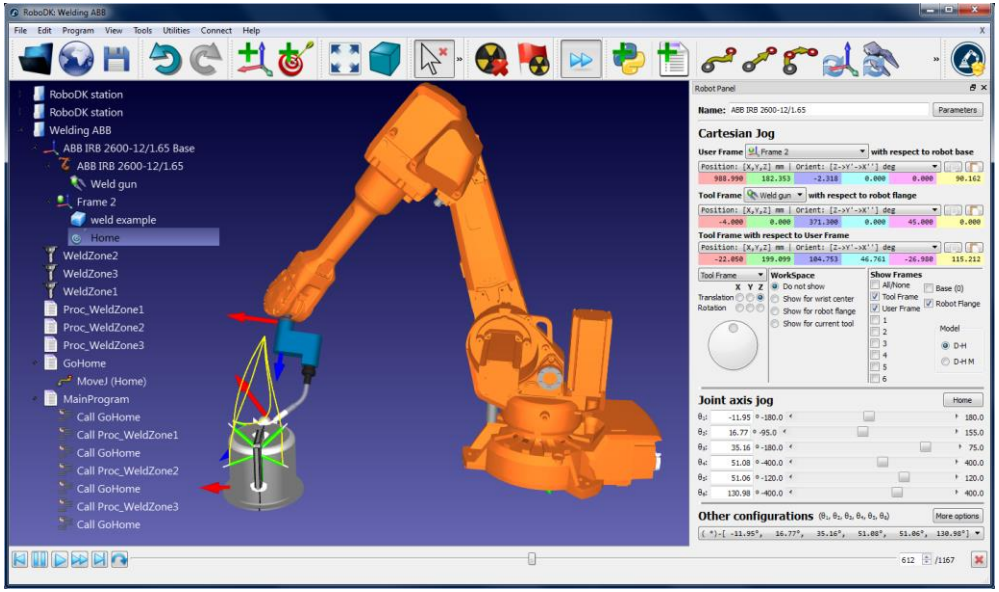
V-REP



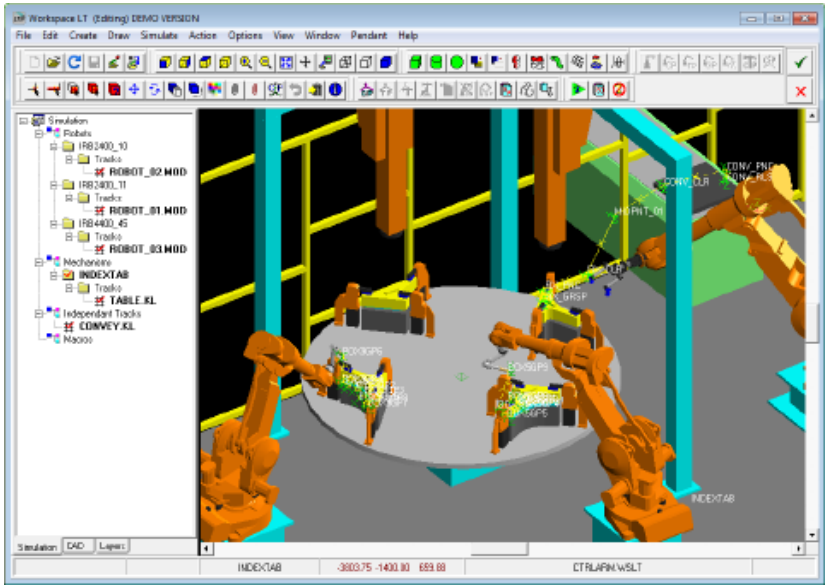
Actin Simulation

4. Mở rộng

4.2. Một số phần mềm mô phỏng Robot



RoboDK



Workspace

. . .

Hết Bài 9