Solid Edge ST7 AddIn EdgeBar in C++

Jason Newell

15 Dec 2014 CPOL

Solid Edge ST7 AddIn EdgeBar in C++

This article was marked as deleted at 10 Mar 2016.

SIEMENS
Siemens PLM Software
SOLID EDGE ST7
© 2014 Siemens Product Lifecycle Management Software Inc.
Image courtesy of Edison2

Table of content

Introduction

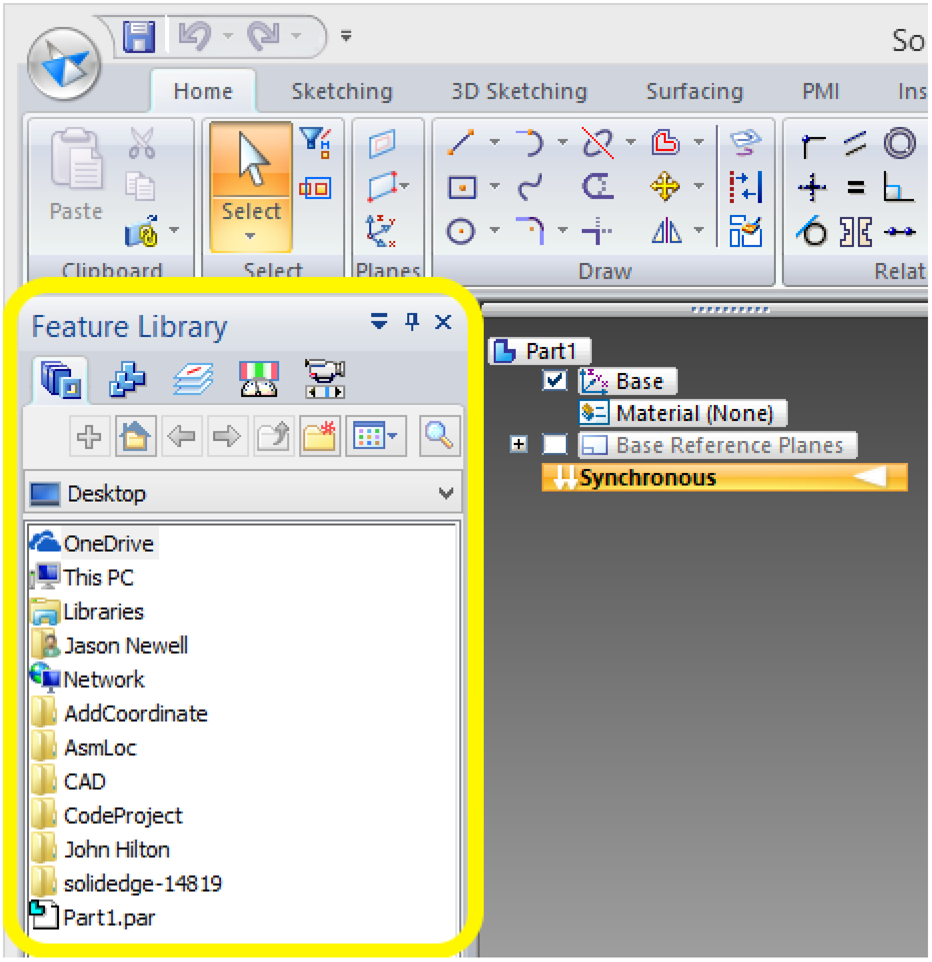
Solid Edge is a 3D CAD application. It is not uncommon for customers or partners to have a need to develop custom code that executes directly inside the Solid Edge process. This article is a continuation of the Solid Edge ST7 AddIn in C++ article and focuses on the bare minimum requirements for implementing an EdgeBar page in a Solid Edge AddIn using C++.

Background

Windows 8.1 x64, Solid Edge ST7 x64 and Visual Studio 2013 Professional was used in creating this article but the steps are mostly the same if you're using different versions. I would also like to point out that there is more than one way to implement an EdgeBar page in a Solid Edge AddIn. In this article, I am simply showing one approach.

What is an EdgeBar?

The Solid Edge EdgeBar is nothing more than a group of docking panes. The Solid Edge API defines these docking panes as "pages". Since Solid Edge is a Multiple Document Interface (MDI) application, each document has its own EdgeBar and corresponding pages. To verbalize the goal of this article, we will be adding a page to the EdgeBar for a given document.

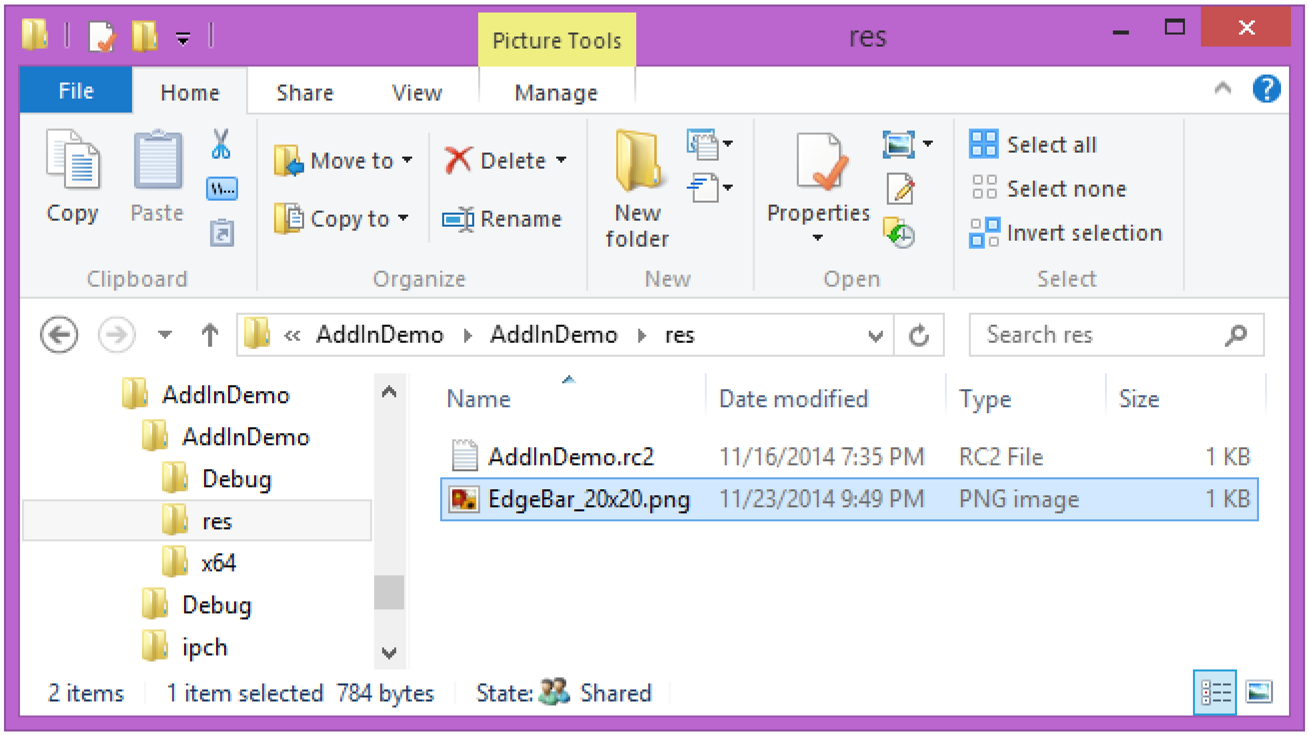


Resources

At a minimum, we will need two resources for our EdgeBar page. A 20x20 PNG or BMP for the tab icon and a dialog that we can use to display inside the page.

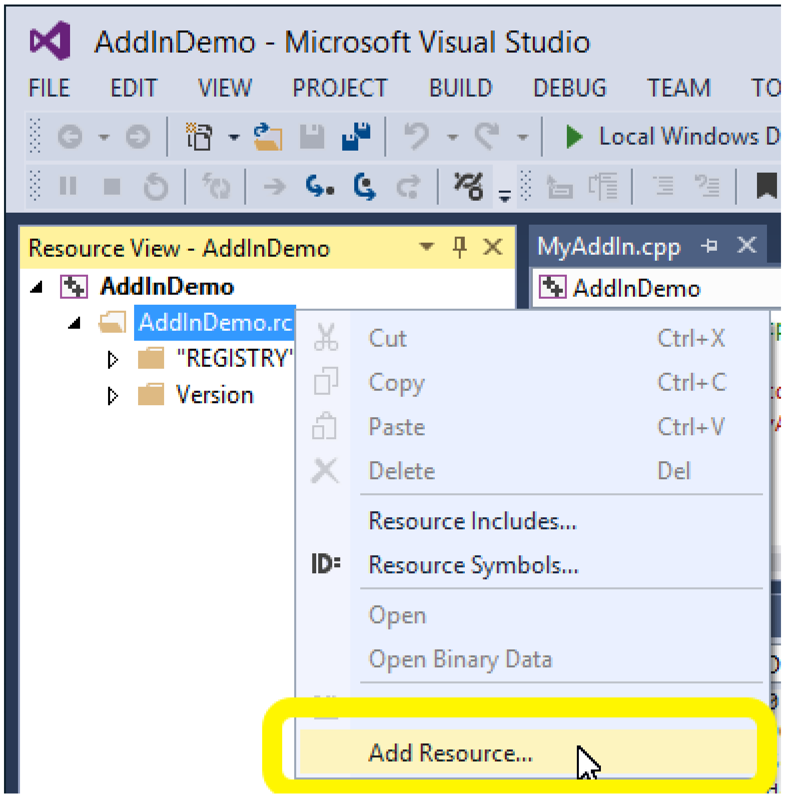
Page icon

Before we can add our EdgeBar page icon to the project, we first need to copy a 20x20 PNG or BMP image to the **\AddInDemo\res** folder.

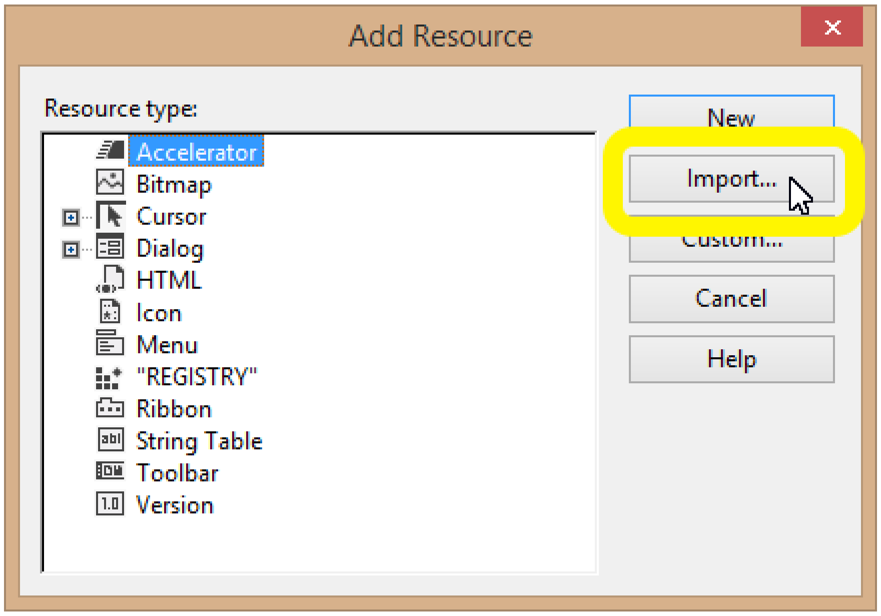


Name	Date modified	Type	Size
AddInDemo.rc2	11/16/2014 7:35 PM	RC2 File	1 KB
EdgeBar_20x20.png	11/23/2014 9:49 PM	PNG image	1 KB

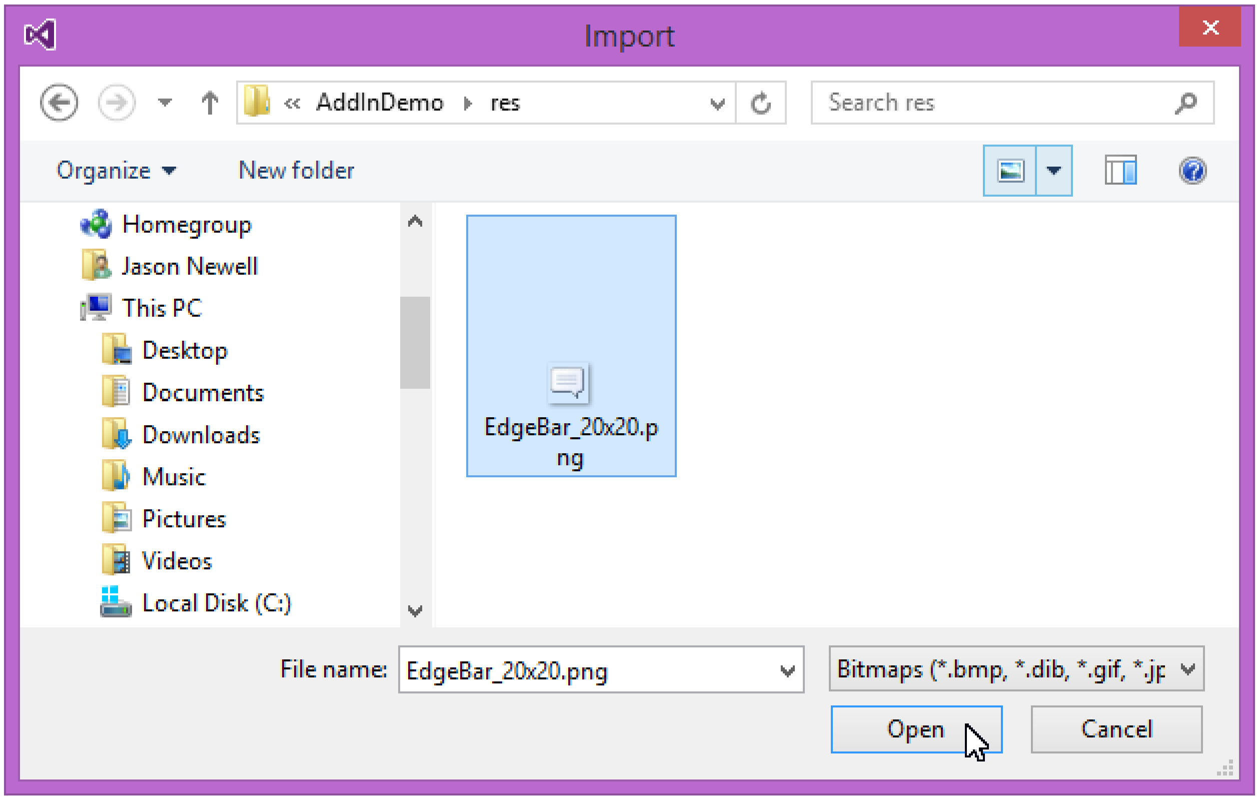
Once the image has been copied to the res folder, navigate to the **Resource View** tab, right click on **AddInDemo.rc** and select **Add Resource...**



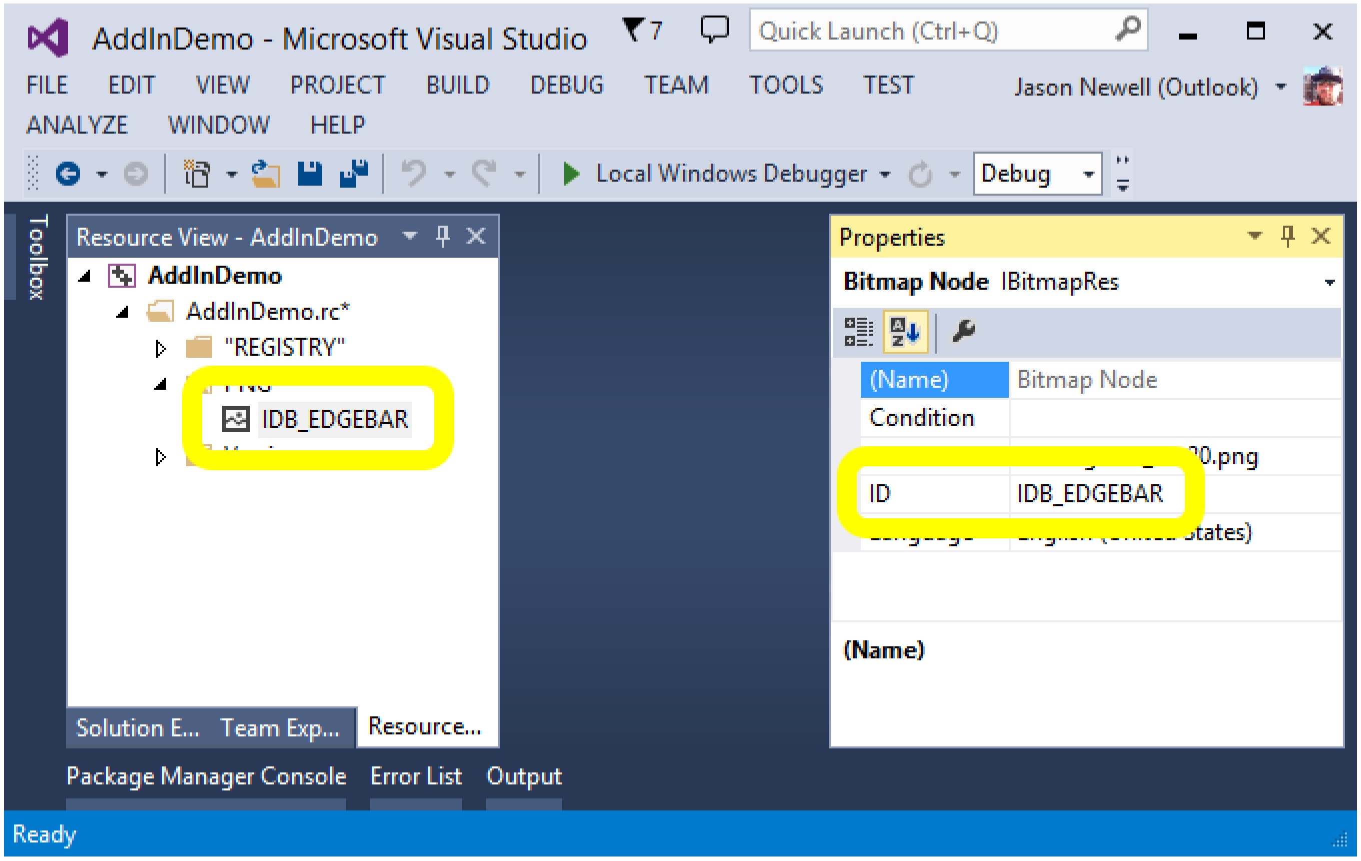
Click the **Import** button.



Navigate to the **\AddInDemo\res** folder and select the PNG\BMP.

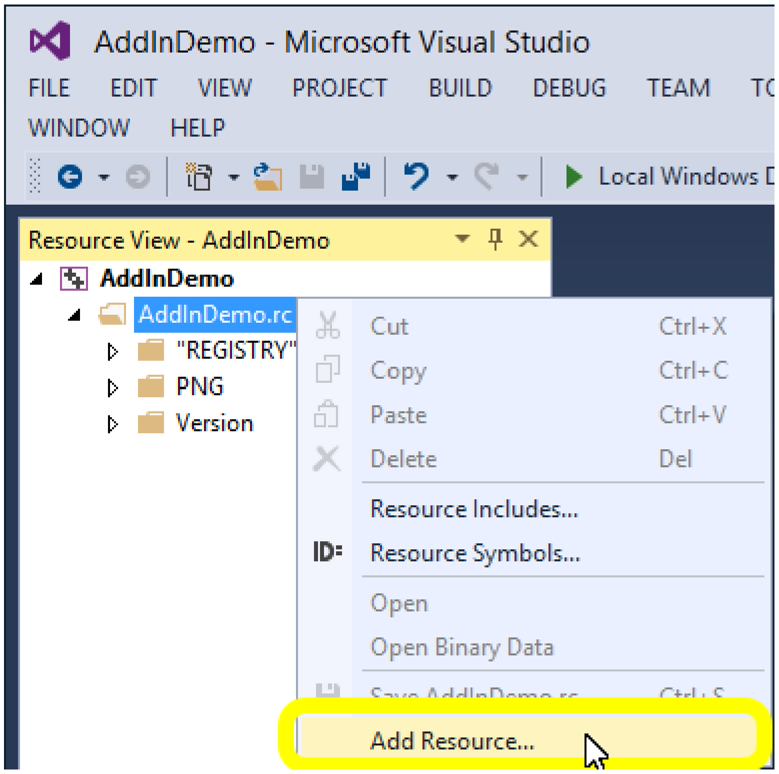


Once the image resource has been added, you can assign it a meaningful **ID**. **IDB_EDGEBAR** was used in this example.

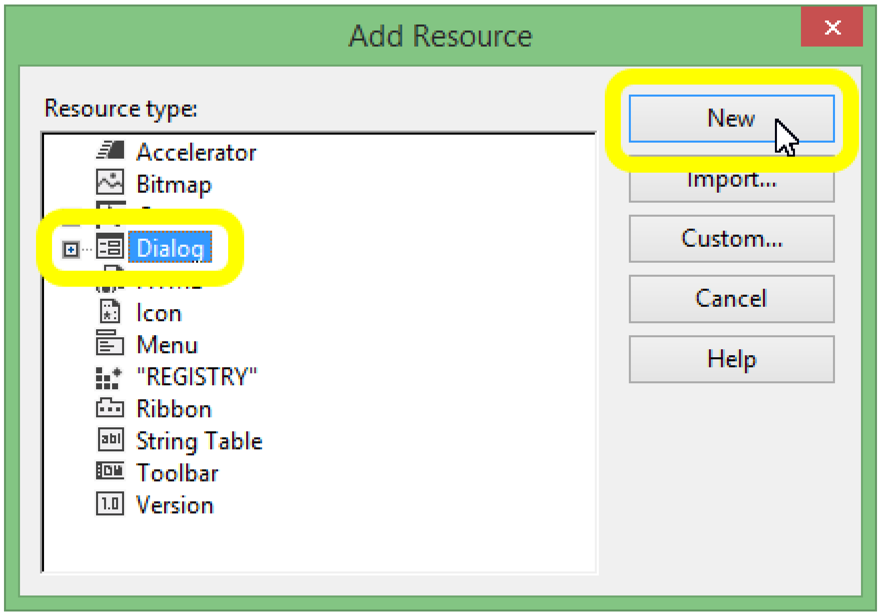

AddInDemo - Microsoft Visual Studio
FILE EDIT VIEW PROJECT BUILD DEBUG TEAM TOOLS TEST ANALYZE WINDOW HELP
Local Windows Debugger
Debug
Resource View - AddInDemo
AddInDemo
AddInDemo.rc*
"REGISTRY"
IDB_EDGEBAR
Properties
Bitmap Node IBitmapRes
(Name) Bitmap Node
Condition
ID IDB_EDGEBAR
(Name)
Solution E... Team Exp... Resource...
Package Manager Console Error List Output
Ready

Page dialog

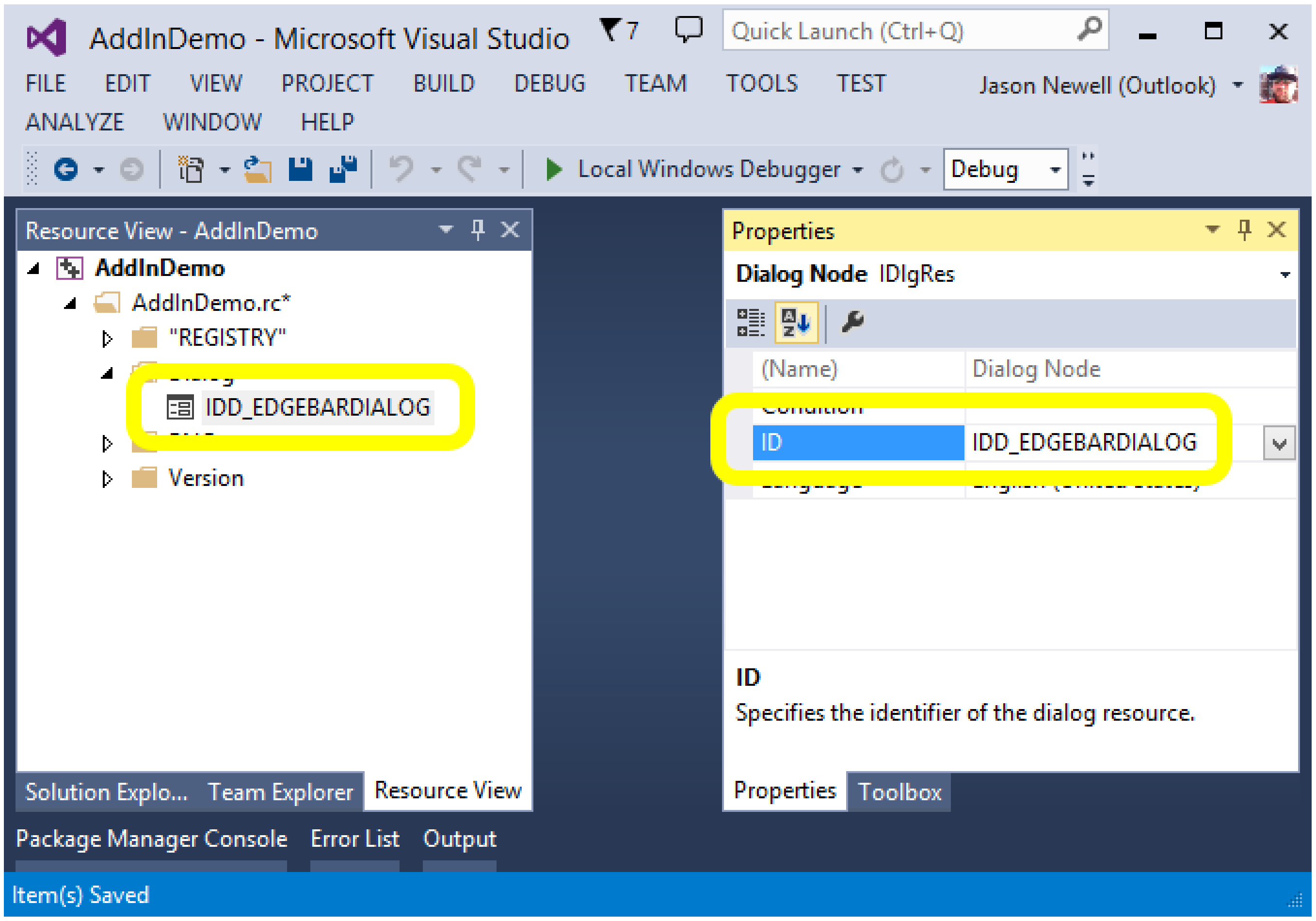
The next step is to create a new dialog resource that will be displayed in the EdgeBar page. In the **Resource View** tab, right click on **AddInDemo.rc** and select **Add Resource...**



Select the resource type **Dialog** and click the **New** button.



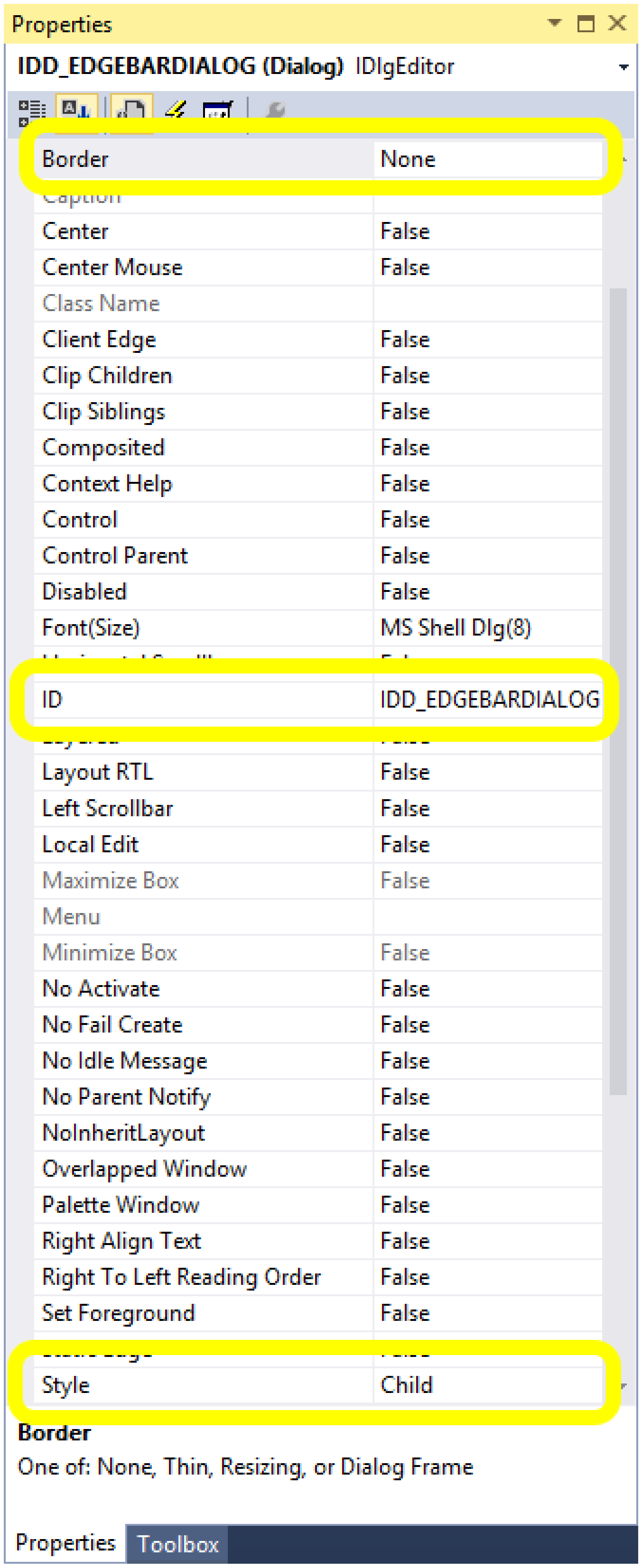
Once the dialog resource has been added, you can assign it a meaningful **ID**. **IDD_EDGEBARDIALOG** was used in this example.


AddInDemo - Microsoft Visual Studio
FILE EDIT VIEW PROJECT BUILD DEBUG TEAM TOOLS TEST
ANALYZE WINDOW HELP
Jason Newell (Outlook)
Local Windows Debugger
Debug
Resource View - AddInDemo
AddInDemo
AddInDemo.rc*
"REGISTRY"
IDD_EDGEBARDIALOG
Version
Properties
Dialog Node IDlgRes
(Name) Dialog Node
ID IDD_EDGEBARDIALOG
ID
Specifies the identifier of the dialog resource.
Solution Explo... Team Explorer Resource View
Properties Toolbox
Package Manager Console Error List Output
Item(s) Saved

Open the dialog in the designer and set the following properties.

- **Border**: None

- **Style**: Child

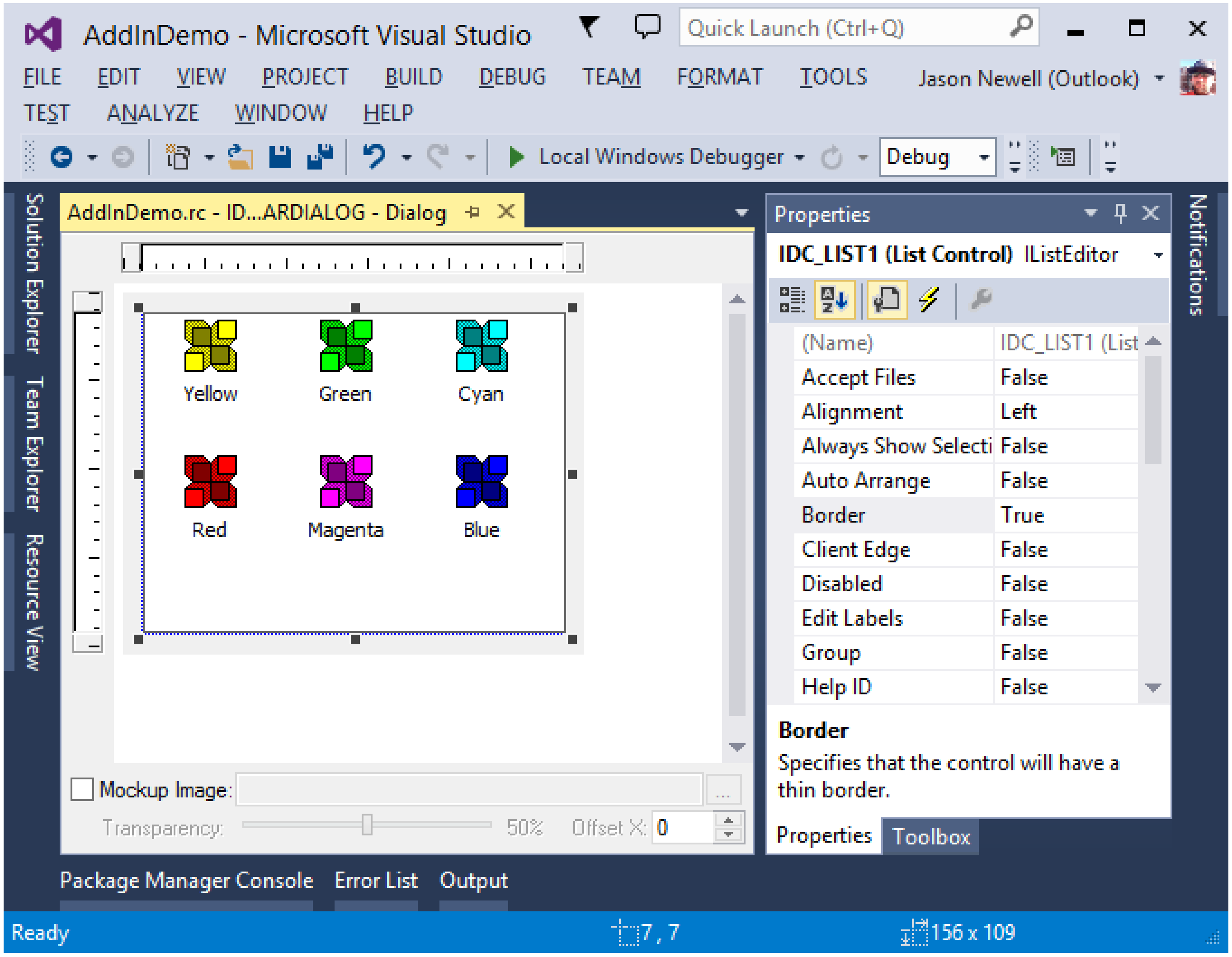

Properties

IDD_EDGEBARDIALOG (Dialog) IDlgEditor

Border	None
Caption	
Center	False
Center Mouse	False
Class Name	
Client Edge	False
Clip Children	False
Clip Siblings	False
Composited	False
Context Help	False
Control	False
Control Parent	False
Disabled	False
Font(Size)	MS Shell Dlg(8)
Horizontal Scrollbar	False
ID	IDD_EDGEBARDIALOG
Layered	False
Layout RTL	False
Left Scrollbar	False
Local Edit	False
Maximize Box	False
Menu	
Minimize Box	False
No Activate	False
No Fail Create	False
No Idle Message	False
No Parent Notify	False
NoInheritLayout	False
Overlapped Window	False
Palette Window	False
Right Align Text	False
Right To Left Reading Order	False
Set Foreground	False
Static Edge	False
Style	Child

Border
One of: None, Thin, Resizing, or Dialog Frame

Remove the **OK** and **Cancel** buttons and add a **List Control** from the **Toolbox**. Optionally, you can name your list in the properties now. In this article, I'll leave it as **IDC_LIST1**.



IDC_LIST1 (List Control) IListEditor

(Name)	IDC_LIST1 (List
Accept Files	False
Alignment	Left
Always Show Selecti	False
Auto Arrange	False
Border	True
Client Edge	False
Disabled	False
Edit Labels	False
Group	False
Help ID	False

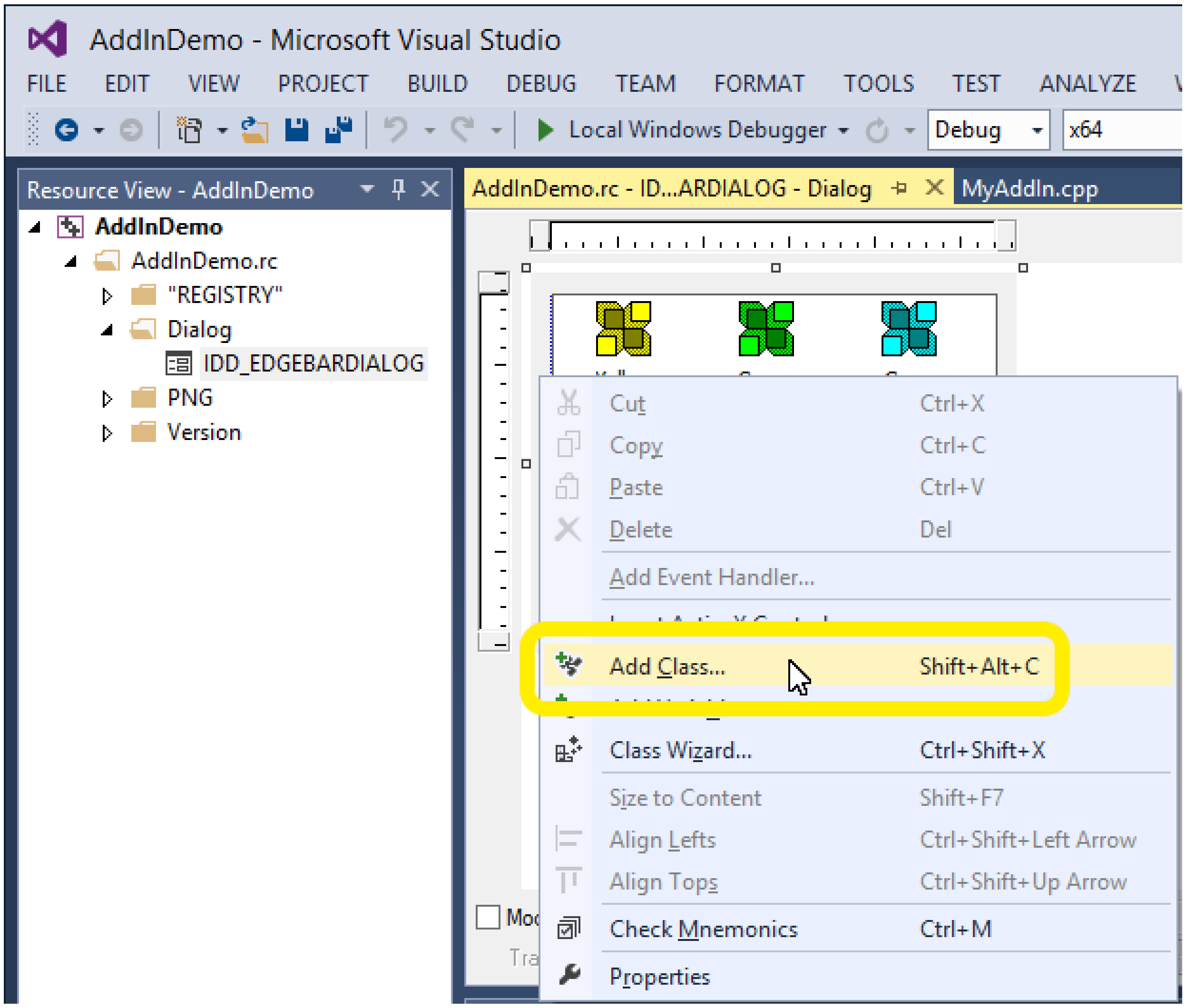
Border
Specifies that the control will have a thin border.

EdgeBar Dialog MFC class

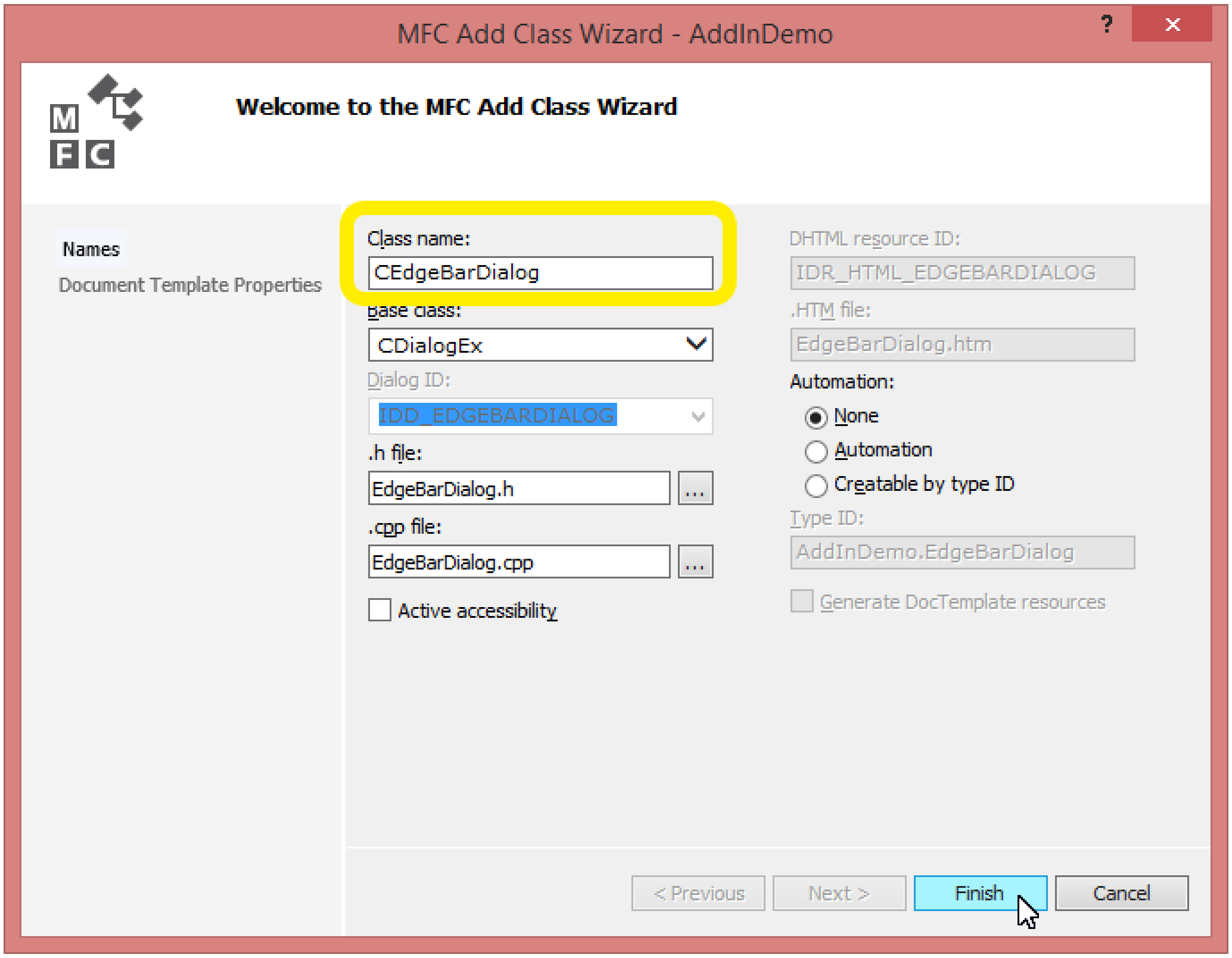
Working with dialogs is made easy with MFC. MFC provides a **CDialogEx** base class that we can inherit from to interact with the dialog.

Creating the dialog MFC class

Now that we have our dialog resource added, we need to create an MFC class to interact with it. From the dialog editor, right click on the dialog and select **Add Class**. This will invoke the MFC Class Wizard for this dialog.


AddInDemo - Microsoft Visual Studio
FILE EDIT VIEW PROJECT BUILD DEBUG TEAM FORMAT TOOLS TEST ANALYZE
Local Windows Debugger
Debug
x64
Resource View - AddInDemo
AddInDemo
AddInDemo.rc
"REGISTRY"
Dialog
IDD_EDGEBARDIALOG
PNG
Version
AddInDemo.rc - ID...ARDIALOG - Dialog
MyAddIn.cpp
Cut Ctrl+X
Copy Ctrl+C
Paste Ctrl+V
Delete Del
Add Event Handler...
Add Class... Shift+Alt+C
Class Wizard... Ctrl+Shift+X
Size to Content Shift+F7
Align Lefts Ctrl+Shift+Left Arrow
Align Tops Ctrl+Shift+Up Arrow
Check Mnemonics Ctrl+M
Properties

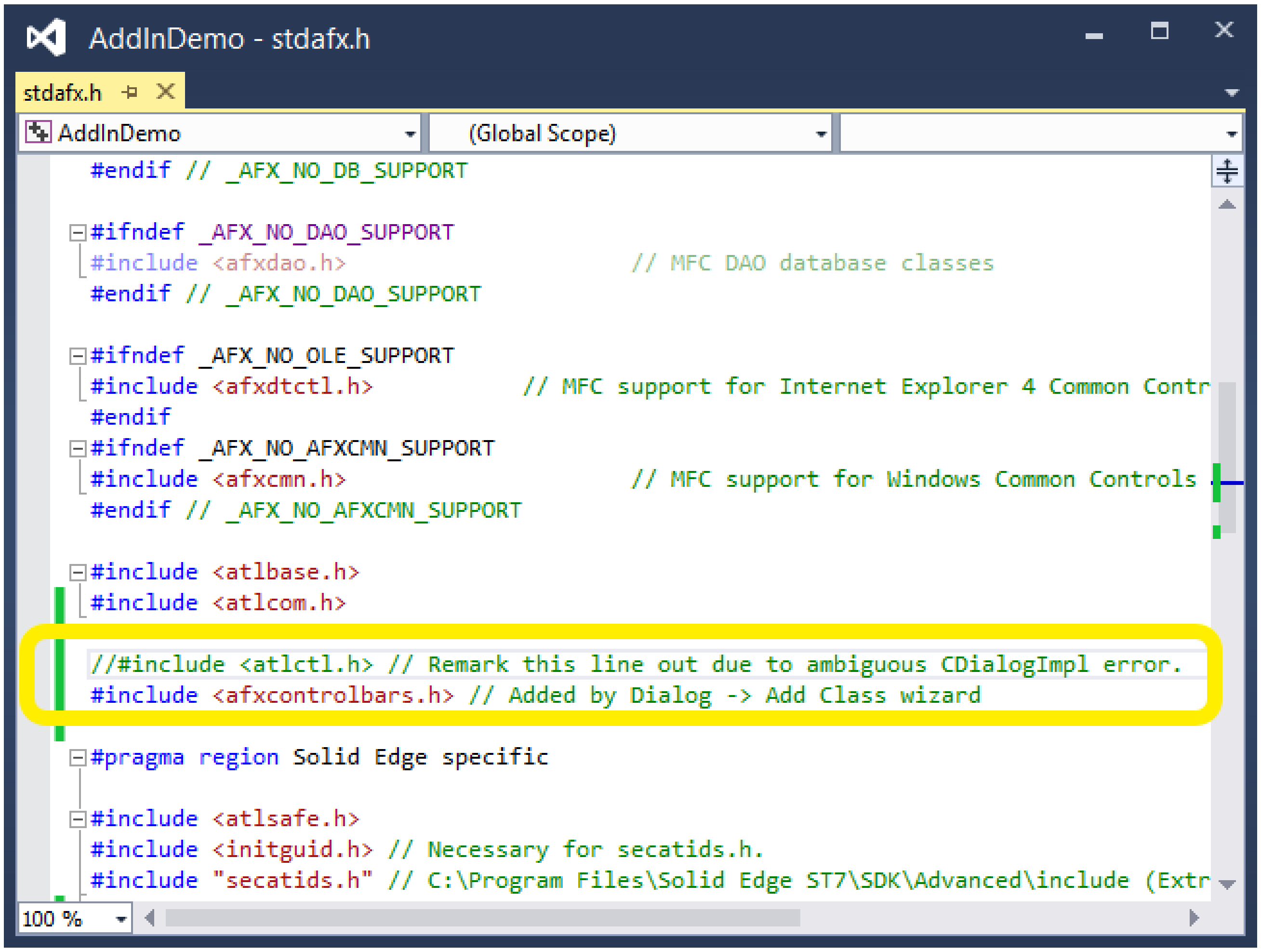
In the MFC Add Class Wizard dialog, enter **CEdgeBarDialog** in the **Class name** field and click Finish.


MFC Add Class Wizard - AddInDemo
Welcome to the MFC Add Class Wizard
Names
Document Template Properties
< Previous
Next >
Finish
Cancel

Class name: CEdgeBarDialog
Base class: CDialogEx
Dialog ID: IDD_EDGEBARDIALOG
.h file: EdgeBarDialog.h
.cpp file: EdgeBarDialog.cpp
Active accessibility: [ ]
DHTML resource ID: IDR_HTML_EDGEBARDIALOG
.HTM file: EdgeBarDialog.htm
Automation: [✓ None] Automation Creatable by type ID
Type ID: AddInDemo.EdgeBarDialog
Generate DocTemplate resources: [ ]

Modifying stdafx.h

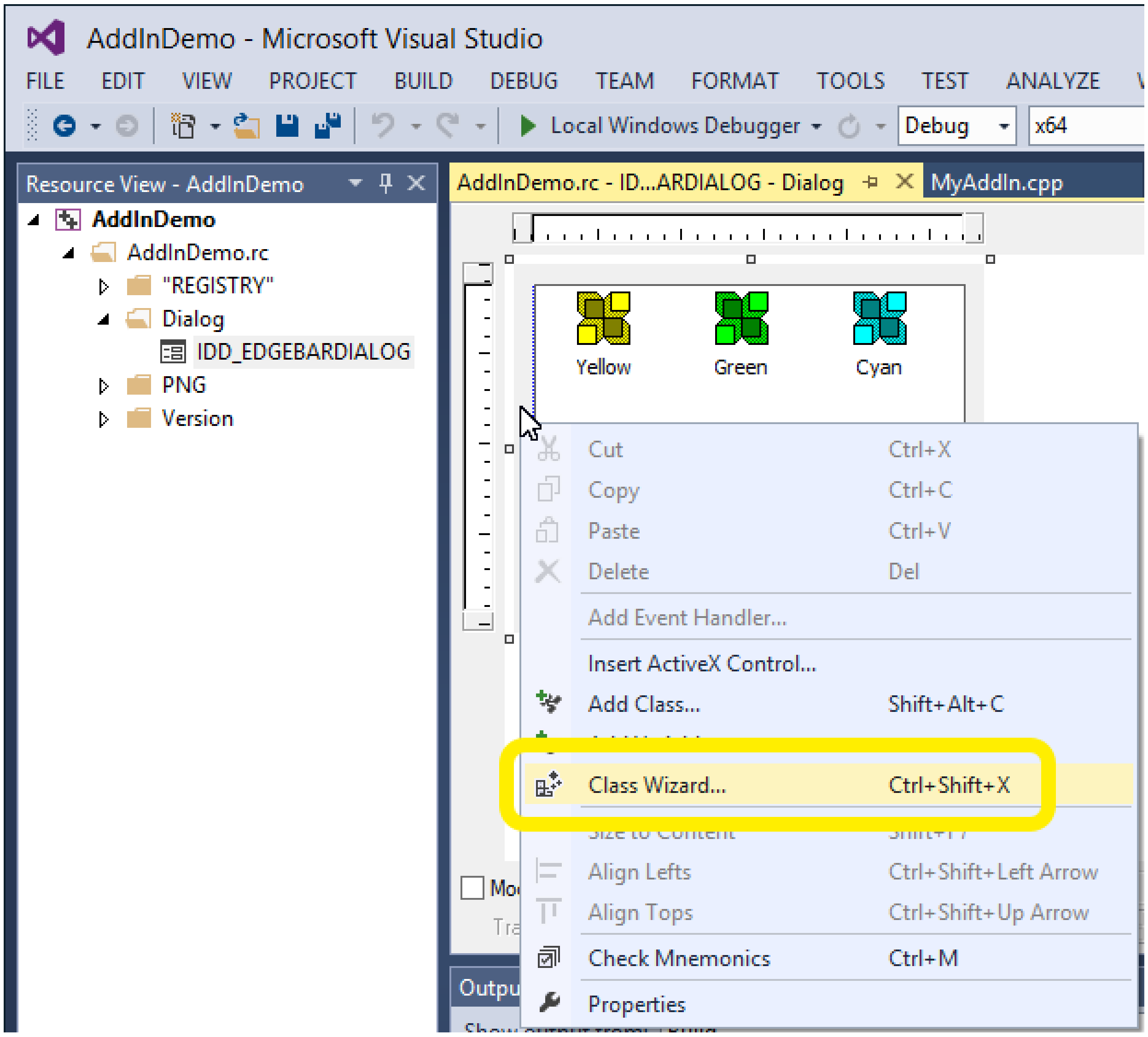
The wizard will now create **EdgeBarDialog.h** and **EdgeBarDialog.cpp**. It also modified **stdafx.h** by adding a new include, **afxcontrolbars.h**. If you try to compile the code right now, you will get a `error C2872: 'CDialogImpl' : ambiguous symbol`. To resolve this, remark out the **atlctl.h** include as shown below.



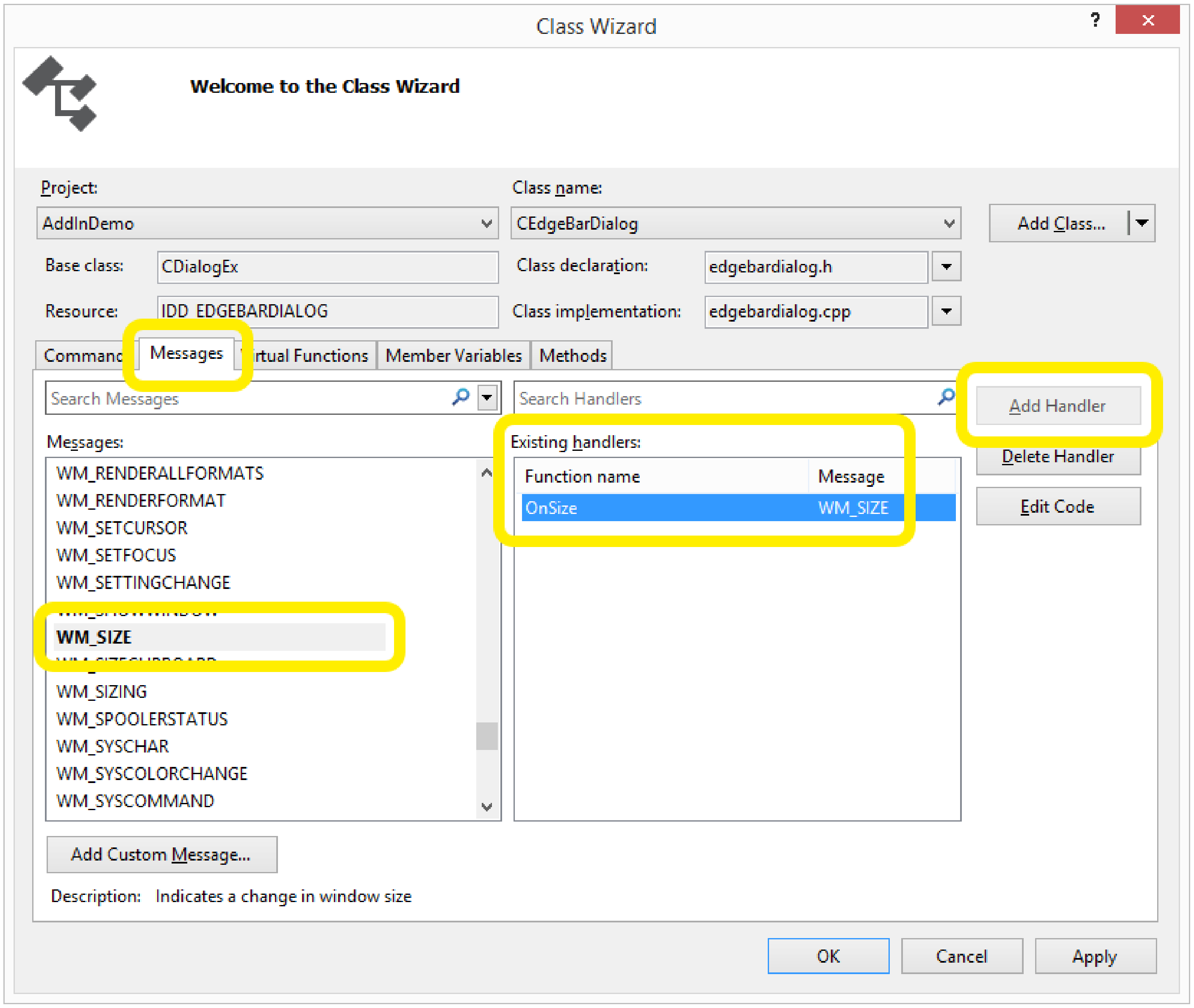
```
#endif // _AFX_NO_DB_SUPPORT

#ifndef _AFX_NO_DAO_SUPPORT
#include <afxdao.h>                     // MFC DAO database classes
#endif // _AFX_NO_DAO_SUPPORT

#ifndef _AFX_NO_OLE_SUPPORT
#include <afxdtctl.h>           // MFC support for Internet Explorer 4 Common Contr
#endif
#ifndef _AFX_NO_AFXCMN_SUPPORT
#include <afxcmn.h>                     // MFC support for Windows Common Controls
#endif // _AFX_NO_AFXCMN_SUPPORT

#include <atlbase.h>
#include <atlcom.h>

//#include <atlctl.h> // Remark this line out due to ambiguous CDialogImpl error.
#include <afxcontrolbars.h> // Added by Dialog -> Add Class wizard

#pragma region Solid Edge specific

#include <atlsafe.h>
#include <initguid.h> // Necessary for secatids.h.
#include "secatids.h" // C:\Program Files\Solid Edge ST7\SDK\Advanced\include (Extr
```

Adding Window message handler

Now that we can compile again, we need to add a `WM_SIZE` message handler to the class. The easy way to do this is to invoke the Class Wizard from the dialog editor.

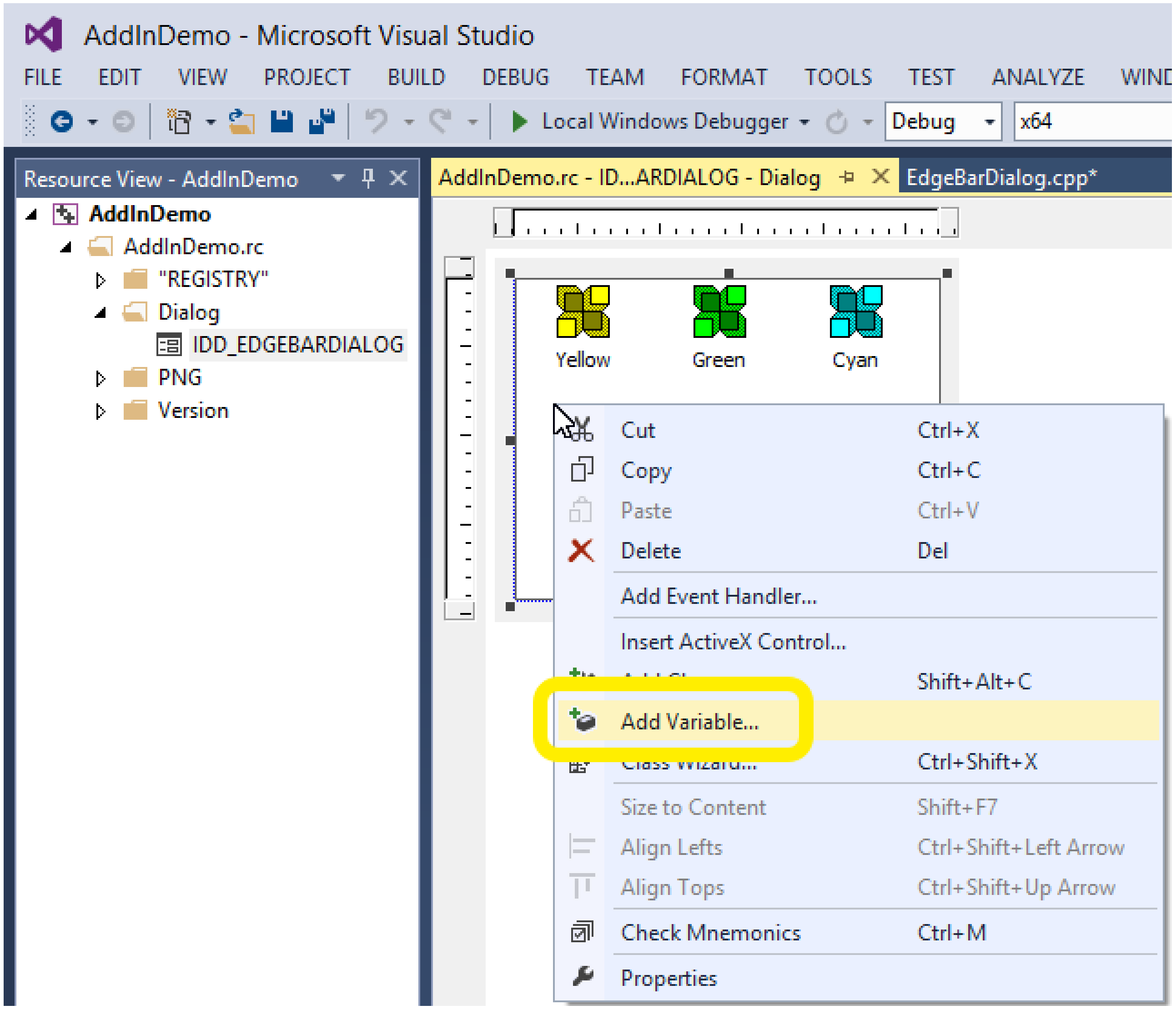

AddInDemo - Microsoft Visual Studio
FILE EDIT VIEW PROJECT BUILD DEBUG TEAM FORMAT TOOLS TEST ANALYZE
Local Windows Debugger
Debug
x64
Resource View - AddInDemo
AddInDemo
AddInDemo.rc
"REGISTRY"
Dialog
IDD_EDGEBARDIALOG
PNG
Version
AddInDemo.rc - ID...ARDIALOG - Dialog
MyAddIn.cpp
Yellow
Green
Cyan
Cut Ctrl+X
Copy Ctrl+C
Paste Ctrl+V
Delete Del
Add Event Handler...
Insert ActiveX Control...
Add Class... Shift+Alt+C
Class Wizard... Ctrl+Shift+X
Align Lefts Ctrl+Shift+Left Arrow
Align Tops Ctrl+Shift+Up Arrow
Check Mnemonics Ctrl+M
Properties

Select the **Messages** tab and scroll until you find **WM_SIZE**. Select **WM_SIZE** and click the **Add Handler** button. Click **OK** to close the dialog.

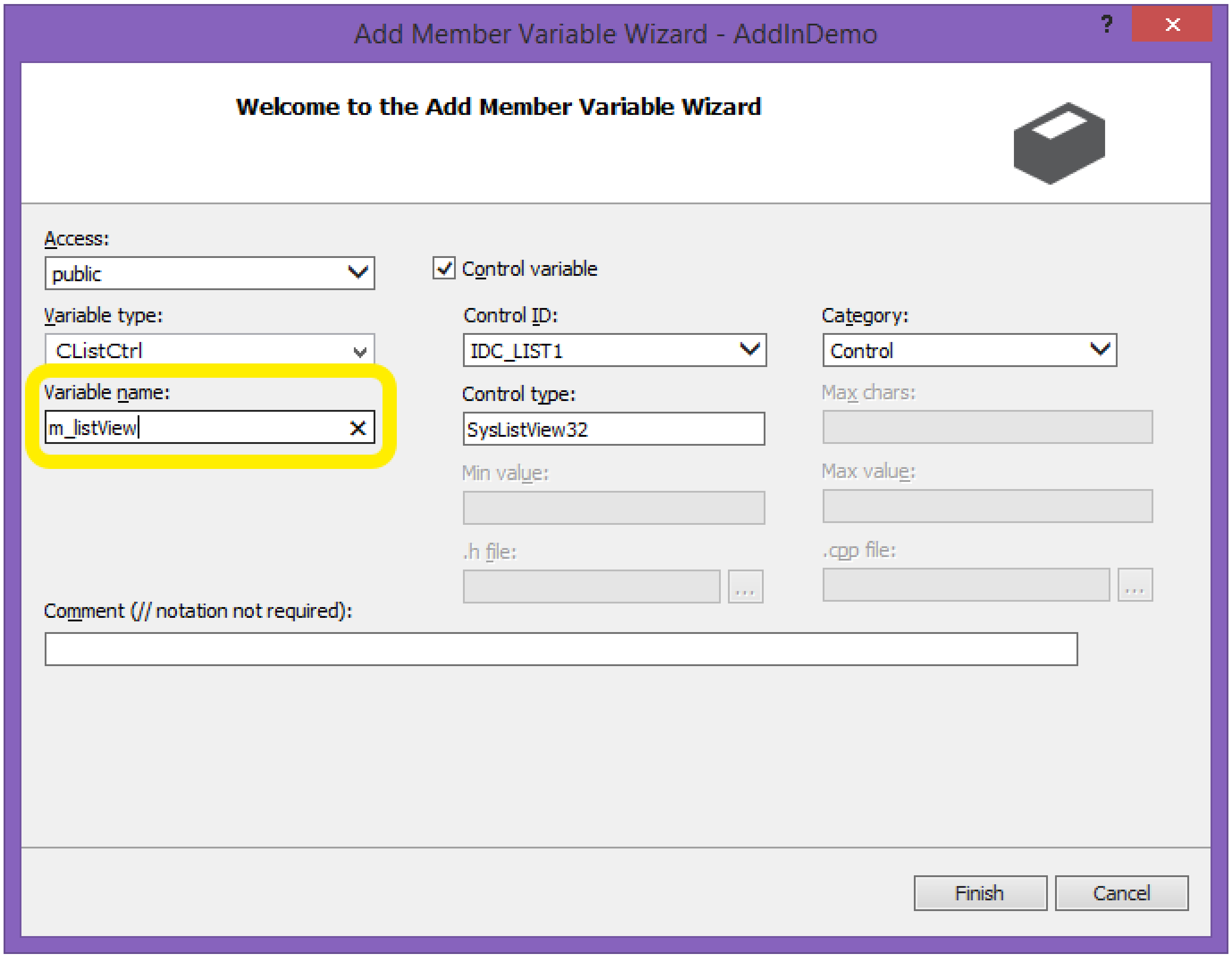


Adding a variable for the List Control

Now we need to add a local variable that will allow us to work with the List Control. From the dialog editor, right click on the List Control and select **Add Variable**.



In the **Variable name** textbox, enter `m_listView`.


Add Member Variable Wizard - AddInDemo
Welcome to the Add Member Variable Wizard
Finish
Cancel

Access: public

☑ Control variable

Variable type: CListCtrl

Control ID: IDC_LIST1

Category: Control

Variable name: m_listView

Control type: SysListView32

Max chars:

Min value:

Max value:

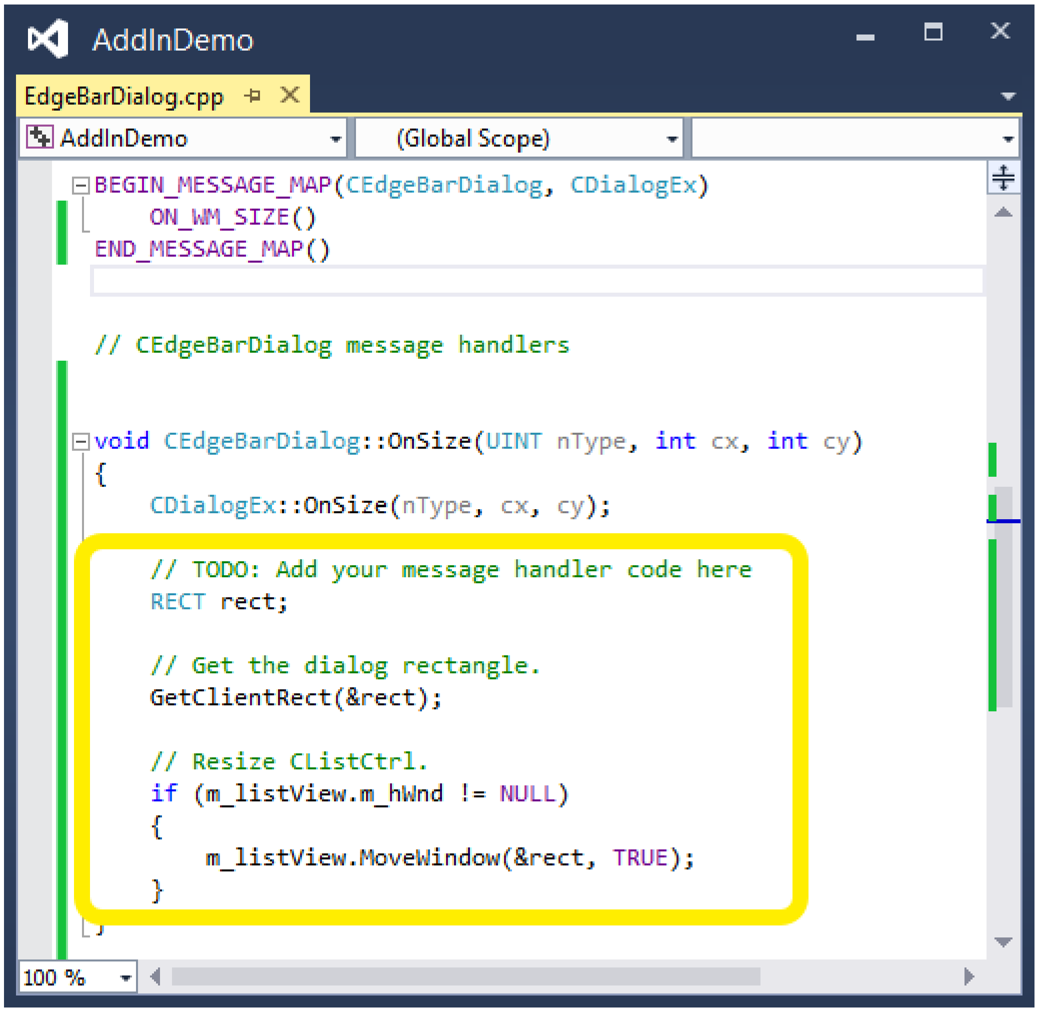
.h file:

.cpp file:

Comment (// notation not required):

Handling WM_SIZE message

Modify **EdgeBarDialog.cpp** so that it will resize our **List Control** when the dialog is resized.



```
BEGIN_MESSAGE_MAP(CEdgeBarDialog, CDialogEx)
    ON_WM_SIZE()
END_MESSAGE_MAP()


// CEdgeBarDialog message handlers


void CEdgeBarDialog::OnSize(UINT nType, int cx, int cy)
{
    CDialogEx::OnSize(nType, cx, cy);

    // TODO: Add your message handler code here
    RECT rect;

    // Get the dialog rectangle.
    GetClientRect(&rect);

    // Resize CListCtrl.
    if (m_listView.m_hWnd != NULL)
    {
        m_listView.MoveWindow(&rect, TRUE);
    }
}
```

EdgeBar Document COM class

Since EdgeBar pages are document specific, we need a COM class that can handle the EdgeBar page and document. You can use the ATL Simple Object Wizard to generate the class but as I've mentioned in previous articles, the wizard adds code that we don't need and have to manually remove. In this example, we will be hand coding the COM class.

Creating EdgeBarDocument.h

Manually create a new header (.h) file, name it **EdgeBarDocument.h** and paste the following code into it.

C++

```
#pragma once

#include "resource.h"
#include "EdgeBarDialog.h"

// {703AADD7-4B5A-41E1-AD20-122FB3588C50}
DEFINE_GUID(CLSID_EdgeBarDocument, 0x703aadd7, 0x4b5a, 0x41e1, 0xad, 0x20, 0x12, 0x2f,
0xb3, 0x58, 0x8c, 0x50);

class CEdgeBarDocument :
    public CComObjectRoot,
    public CComCoClass<CEdgeBarDocument, &CLSID_EdgeBarDocument>
{
protected:
    SolidEdgeDocumentPtr m_pDocument;
    CWnd m_hWndEdgeBarPage;
    CEdgeBarDialog* m_pDialog;

public:
    CEdgeBarDocument();
    ~CEdgeBarDocument();

    HRESULT CreateEdgeBarPage(ISolidEdgeBarEx* pEdgeBarEx, SolidEdgeDocument* pDocument);
    HRESULT DeleteEdgeBarPage(ISolidEdgeBarEx* pEdgeBarEx, SolidEdgeDocument* pDocument);

    BEGIN_COM_MAP(CEdgeBarDocument)
    END_COM_MAP()
    DECLARE_NOT_AGGREGATABLE(CEdgeBarDocument)
};

typedef CComObject<CEdgeBarDocument> CEdgeBarDocumentObj;
```

Creating EdgeBarDocument.cpp

Manually create a new C++ (.cpp) file, name it **EdgeBarDocument.cpp** and paste the following code into it. See code comments for details.

The two relevant methods of this class are **CreateEdgeBarPage** and **DeleteEdgeBarPage**. The **CreateEdgeBarPage** method interacts with the Solid Edge API to create a new EdgeBar page and create a new instance of the dialog. The **DeleteEdgeBarPage** method interacts with the Solid Edge API to remove the previously added EdgeBar page and delete the dialog.

C++

```
#include "stdafx.h"
#include "EdgeBarDocument.h"
#include "resource.h"

CEdgeBarDocument::CEdgeBarDocument()
{
    m_pDocument = NULL;
}

CEdgeBarDocument::~CEdgeBarDocument()
{
    m_pDocument = NULL;
}

HRESULT CEdgeBarDocument::CreateEdgeBarPage(ISolidEdgeBarEx* pEdgeBarEx, SolidEdgeDocument*
pDocument)
{
    m_pDocument = pDocument;

    // Build path to .dll that contains the resources.
    HINSTANCE hInstance = AfxGetResourceHandle();
    TCHAR ResourceFilename[MAX_PATH];
    GetModuleFileName(hInstance, ResourceFilename, sizeof(ResourceFilename));

    CString strTooltip;
    strTooltip = L"AddInDemo";
    _bstr_t bstrTooltip = strTooltip;

    // Create a new EdgeBar page.
    HWND hWndEdgeBarPage = (HWND)pEdgeBarEx->AddPageEx(pDocument, ResourceFilename,
IDB_EDGEBAR, bstrTooltip, 2);

    if (hWndEdgeBarPage)
    {
        // Attach a CWnd local variable to the new EdgeBar page.
        BOOL bAttached = m_hWndEdgeBarPage.Attach(hWndEdgeBarPage);

        if (bAttached)
        {
            // Create a new dialog and make it a child of the new EdgeBar page.
            m_pDialog = new CEdgeBarDialog();
            m_pDialog->Create(IDD_EDGEBARDIALOG, &m_hWndEdgeBarPage);
            m_pDialog->ShowWindow(SW_SHOW);
        }
    }

    return S_OK;
}

HRESULT CEdgeBarDocument::DeleteEdgeBarPage(ISolidEdgeBarEx* pEdgeBarEx, SolidEdgeDocument*
pDocument)
{
    HRESULT hr = S_OK;

    // Detach the CWnd and obtain the HWND.
    HWND hWndEdgeBarPage = m_hWndEdgeBarPage.Detach();

    // Remove the EdgeBar page.
    hr = pEdgeBarEx->RemovePage(pDocument, (LONG)hWndEdgeBarPage, 0);

    // Cleanup dialog.
    delete m_pDialog;
    m_pDialog = NULL;

    return S_OK;
}
```

EdgeBar Controller COM class

The Solid Edge API provides us with the `ISEAddInEdgeBarEvents` interface that will notify us when it's time to add or remove an EdgeBar page to a specific document. There are obviously various ways to implement handling COM events but in this article, I chose to implement a controller type class.

Creating XEventHandler.h

Before you can sink COM events, you needs a way to do so. An easy way to accomplish this is to create a C++ template class that can be used in any class that you need to sink events.

Manually create a new header (.h) file, name it **XEventHandler.h** and paste the following code into it.

C++

```
#pragma once
#include "stdafx.h"

template <class IEvents, const IID* piidEvents, const GUID* plibid, class XEvents, const
CLSID* pClsidEvents>
class XEventHandler :
    public CComDualImpl<IEvents, piidEvents, plibid>,
    public CComObjectRoot,
    public CComCoClass<XEvents, pClsidEvents>
{
public:
    BEGIN_COM_MAP(XEvents)
        COM_INTERFACE_ENTRY_IID(*piidEvents, IEvents)
    END_COM_MAP()

    DECLARE_NOT_AGGREGATABLE(XEvents)

    HRESULT Connect(IUnknown* pUnk)
    {
        HRESULT hr; VERIFY(SUCCEEDED(hr = AtlAdvise(pUnk, this, *piidEvents,
&m_dwAdvise))); return hr;
    }

    HRESULT Disconnect(IUnknown* pUnk)
    {
        HRESULT hr; VERIFY(SUCCEEDED(hr = AtlUnadvise(pUnk, *piidEvents, m_dwAdvise)));
return hr;
    }

protected:
    DWORD m_dwAdvise;
};
```

Creating EdgeBarController.h

Manually create a new header (.h) file, name it **EdgeBarController.h** and paste the following code into it.

Note that the **CEdgeBarController** class has a nested class, **XAddInEdgeBarEvents**. The **XAddInEdgeBarEvents** nested class inherits from **XEventHandler** to implement the **ISEAddInEdgeBarEvents** interface. Events received by **XAddInEdgeBarEvents** are forwarded to **CEdgeBarController.**

C++

```cpp
#pragma once
#include "AddInDemo_i.h"
#include "XEventHandler.h"
#include "EdgeBarDocument.h"

// {AC9C8327-445B-4C98-81ED-D96E60A0F7E2}
DEFINE_GUID(CLSID_EdgeBarController, 0xac9c8327, 0x445b, 0x4c98, 0x81, 0xed, 0xd9, 0x6e,
0x60, 0xa0, 0xf7, 0xe2);

// {52BB414E-E570-40F0-8148-6FCD43201F00}
DEFINE_GUID(CLSID_XAddInEdgeBarEvents, 0x52bb414e, 0xe570, 0x40f0, 0x81, 0x48, 0x6f, 0xcd,
0x43, 0x20, 0x1f, 0x0);

// Define a mapping from a Solid Edge document dispatch pointer to my add-in document.
typedef CTypedPtrMap<CMapPtrToPtr, SolidEdgeDocument*, CComObject<CEdgeBarDocument>*>
CMapDocumentToEdgeBarPage;

// Local COM class that handles adding\removing EdgeBar pages.
class CEdgeBarController :
    public CComObjectRoot,
    public CComCoClass<CEdgeBarController, &CLSID_EdgeBarController>
{
public:
    CEdgeBarController();
    ~CEdgeBarController();

    HRESULT SetAddInEx(ISEAddInEx* pAddInEx);

    BEGIN_COM_MAP(CEdgeBarController)
    END_COM_MAP()
    DECLARE_NOT_AGGREGATABLE(CEdgeBarController)

protected:
    CMapDocumentToEdgeBarPage m_pMap;
    ISEAddInExPtr m_pAddInEx;

    HRESULT AddEdgeBarPage(SolidEdgeDocument* pDocument);
    HRESULT RemoveEdgeBarPage(SolidEdgeDocument* pDocument);
    HRESULT RemoveAllEdgeBarPages();

protected:
    // Nested COM class that handles ISEAddInEdgeBarEvents.
    class XAddInEdgeBarEvents : public XEventHandler < ISEAddInEdgeBarEvents,
        &__uuidof(ISEAddInEdgeBarEvents),
        &LIBID_AddInDemoLib,
        XAddInEdgeBarEvents,
        &CLSID_XAddInEdgeBarEvents >
    {
    public:
        // ISEAddInEvents methods
        STDMETHOD(raw_AddPage) (LPDISPATCH theDocument);
        STDMETHOD(raw_RemovePage) (LPDISPATCH theDocument);
        STDMETHOD(raw_IsPageDisplayable) (LPDISPATCH theDocument, BSTR EnvironmentCatID,
VARIANT_BOOL * vbIsPageDisplayable);

        CEdgeBarController* m_pController;
    };

    typedef CComObject<XAddInEdgeBarEvents> XAddInEdgeBarEventsObj;
    XAddInEdgeBarEventsObj* m_pAddInEdgeBarEventsObj;
};

typedef CComObject<CEdgeBarController> CEdgeBarControllerObj;
```

Creating EdgeBarController.cpp

Manually create a new header (.h) file, name it **EdgeBarController.cpp** and paste the following code into it. See code comments for details.

C++

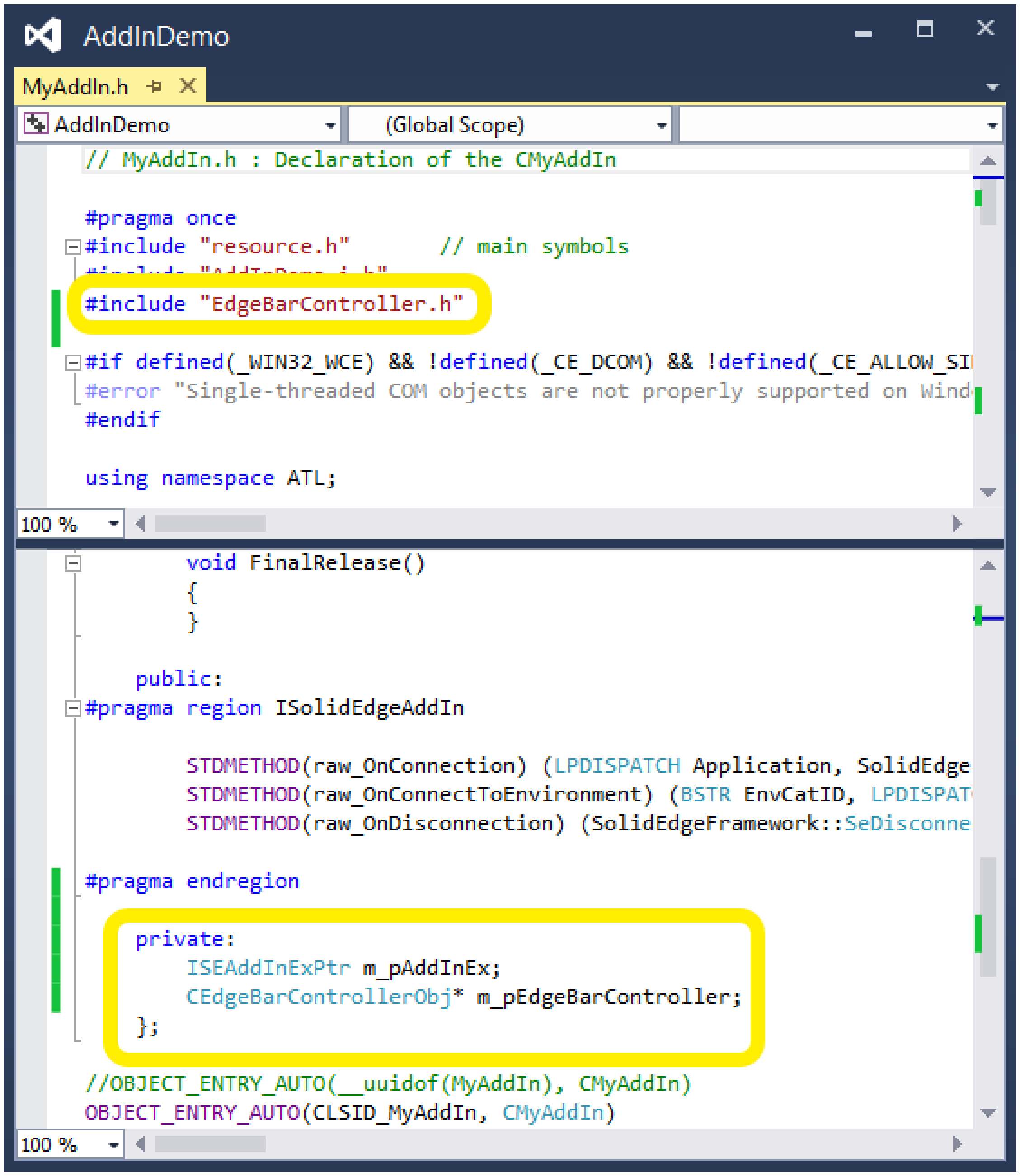
```
#include "stdafx.h"
#include "EdgeBarController.h"

CEdgeBarController::CEdgeBarController()
{
    m_pAddInEx = NULL;
    m_pAddInEdgeBarEventsObj = NULL;
}

CEdgeBarController::~CEdgeBarController()
{
    m_pAddInEdgeBarEventsObj = NULL;
    m_pAddInEx = NULL;
}

HRESULT CEdgeBarController::SetAddInEx(ISEAddInEx* pAddInEx)
{
    HRESULT hr = S_OK;
    SolidEdgeFramework::ApplicationPtr pApplication = NULL;
    SolidEdgeFramework::SolidEdgeDocumentPtr pDocument = NULL;

    // If SetAddInEx() has been previously called, disconnect from AddInEdgeBarEvents.
    if (m_pAddInEx != NULL)
    {
        hr = RemoveAllEdgeBarPages();

        if (m_pAddInEdgeBarEventsObj != NULL)
        {
            hr = m_pAddInEdgeBarEventsObj->Disconnect(m_pAddInEx->AddInEdgeBarEvents);
            m_pAddInEdgeBarEventsObj->Release();
            m_pAddInEdgeBarEventsObj = NULL;
        }
    }

    // Assign local variable.
    m_pAddInEx = pAddInEx;

    // If m_pAddInEx is not NULL, connect to AddInEdgeBarEvents.
    if (m_pAddInEx != NULL)
    {
        XAddInEdgeBarEventsObj::CreateInstance(&m_pAddInEdgeBarEventsObj);

        if (m_pAddInEdgeBarEventsObj != NULL)
        {
            m_pAddInEdgeBarEventsObj->AddRef();
            hr = m_pAddInEdgeBarEventsObj->Connect(m_pAddInEx->AddInEdgeBarEvents);
            m_pAddInEdgeBarEventsObj->m_pController = this;
        }

        pApplication = m_pAddInEx->Application;

        // ActiveDocument property will throw an exception if no document is open.
        _ATLTRY
        {
            pDocument = pApplication->ActiveDocument;

            if (pDocument != NULL)
            {
                AddEdgeBarPage(pDocument);
            }
        }
            _ATLCATCHALL()
        {
        }
    }

    pDocument = NULL;
    pApplication = NULL;

    return S_OK;
}

HRESULT CEdgeBarController::AddEdgeBarPage(SolidEdgeDocument* pDocument)
{
    AFX_MANAGE_STATE(AfxGetStaticModuleState());

    HRESULT hr = S_OK;
    CEdgeBarDocumentObj* pEdgeBarDocument = NULL;
    ISolidEdgeBarExPtr pEdgeBarEx = NULL;

    // Check our map to see if we've already created an EdgeBar page for the document.
    BOOL bFound = m_pMap.Lookup(pDocument, pEdgeBarDocument);

    if (bFound == FALSE)
    {
        // Get a pointer to the ISolidEdgeBarEx interface.
        pEdgeBarEx = m_pAddInEx;

        // Create an instance of local COM object CEdgeBarDocument.
        CEdgeBarDocumentObj::CreateInstance(&pEdgeBarDocument);

        // Manually AddRef() it.
        pEdgeBarDocument->AddRef();

        // EdgeBar page creation logic is contained in the CEdgeBarDocument.
        hr = pEdgeBarDocument->CreateEdgeBarPage(pEdgeBarEx, pDocument);

        if (SUCCEEDED(hr))
        {
            // Add it to our map so we can keep track of which documents we've added
EdgeBar pages to.
            m_pMap.SetAt(pDocument, pEdgeBarDocument);
        }
    }

    pEdgeBarEx = NULL;
    return S_OK;
}

HRESULT CEdgeBarController::RemoveEdgeBarPage(SolidEdgeDocument* pDocument)
{
    AFX_MANAGE_STATE(AfxGetStaticModuleState());

    HRESULT hr = S_OK;
    CEdgeBarDocumentObj* pEdgeBarDocument = NULL;
    ISolidEdgeBarExPtr pEdgeBarEx = NULL;

    // Check our map to see if we've created an EdgeBar page for the document.
    BOOL bFound = m_pMap.Lookup(pDocument, pEdgeBarDocument);

    if ((bFound == TRUE) && (pEdgeBarDocument != NULL))
    {
        // Remove the entry from the map.
        m_pMap.RemoveKey(pDocument);

        // Get a pointer to the ISolidEdgeBarEx interface.
        pEdgeBarEx = m_pAddInEx;

        // Give the CEdgeBarDocument the opportunity to clean up.
        hr = pEdgeBarDocument->DeleteEdgeBarPage(pEdgeBarEx, pDocument);

        // This is the final release.
        pEdgeBarDocument->Release();
    }

    pEdgeBarEx = NULL;
    return S_OK;
}

HRESULT CEdgeBarController::RemoveAllEdgeBarPages()
{
    AFX_MANAGE_STATE(AfxGetStaticModuleState());

    HRESULT hr = S_OK;
    CEdgeBarDocumentObj* pEdgeBarDocument = NULL;
    ISolidEdgeBarExPtr pEdgeBarEx = NULL;
    SolidEdgeDocument* pDocument = NULL;

    // Get a pointer to the ISolidEdgeBarEx interface.
    pEdgeBarEx = m_pAddInEx;

    POSITION position = m_pMap.GetStartPosition();
    while (position)
    {
        m_pMap.GetNextAssoc(position, pDocument, pEdgeBarDocument);

        if (pEdgeBarDocument != NULL)
        {
            // Give the CEdgeBarDocument the opportunity to clean up.
            hr = pEdgeBarDocument->DeleteEdgeBarPage(pEdgeBarEx, pDocument);

            // This is the final release.
            pEdgeBarDocument->Release();
        }
    }

    m_pMap.RemoveAll();

    pEdgeBarEx = NULL;
    return S_OK;
}

HRESULT CEdgeBarController::XAddInEdgeBarEvents::raw_AddPage(LPDISPATCH theDocument)
{
    HRESULT hr = S_OK;
    SolidEdgeDocumentPtr pDocument = theDocument;

    if (pDocument != NULL)
    {
        // Forward call to controller.
        hr = m_pController->AddEdgeBarPage(pDocument);
    }
    else
    {
        hr = E_FAIL;
    }

    pDocument = NULL;
    return hr;
}

HRESULT CEdgeBarController::XAddInEdgeBarEvents::raw_RemovePage(LPDISPATCH theDocument)
{
    HRESULT hr = S_OK;
    SolidEdgeDocumentPtr pDocument = theDocument;

    if (pDocument != NULL)
    {
        // Forward call to controller.
        hr = m_pController->RemoveEdgeBarPage(pDocument);
    }
    else
    {
        hr = E_FAIL;
    }
    pDocument = NULL;
    return hr;
}

HRESULT CEdgeBarController::XAddInEdgeBarEvents::raw_IsPageDisplayable(LPDISPATCH
theDocument, BSTR EnvironmentCatID, VARIANT_BOOL * vbIsPageDisplayable)
{
    *vbIsPageDisplayable = VARIANT_TRUE;
    return S_OK;
}
```

Putting it all together

Now that everything is in place, all we need to do is modify our addin class to use the CEdgeBarController. Since the EdgeBar logic is encapsulated in CEdgeBarController, our addin class logic is short and straightforward.

Modify MyAddIn.h

Modify your MyAddIn.h file as shown below. Here we include EdgeBarController.h and add a couple of private variables.



```
// MyAddIn.h : Declaration of the CMyAddIn

#pragma once
#include "resource.h"       // main symbols
#include "AddInDemo_i.h"
#include "EdgeBarController.h"

#if defined(_WIN32_WCE) && !defined(_CE_DCOM) && !defined(_CE_ALLOW_SI
#error "Single-threaded COM objects are not properly supported on Wind
#endif

using namespace ATL;
```

```
        void FinalRelease()
        {
        }

    public:
#pragma region ISolidEdgeAddIn

        STDMETHOD(raw_OnConnection) (LPDISPATCH Application, SolidEdge
        STDMETHOD(raw_OnConnectToEnvironment) (BSTR EnvCatID, LPDISPAT
        STDMETHOD(raw_OnDisconnection) (SolidEdgeFramework::SeDisconne

#pragma endregion

    private:
        ISEAddInExPtr m_pAddInEx;
        CEdgeBarControllerObj* m_pEdgeBarController;
    };

//OBJECT_ENTRY_AUTO(__uuidof(MyAddIn), CMyAddIn)
OBJECT_ENTRY_AUTO(CLSID_MyAddIn, CMyAddIn)
```

Modify MyAddIn.cpp

Modify your **MyAddIn.cpp** and paste the following code into it. See code comments for details.

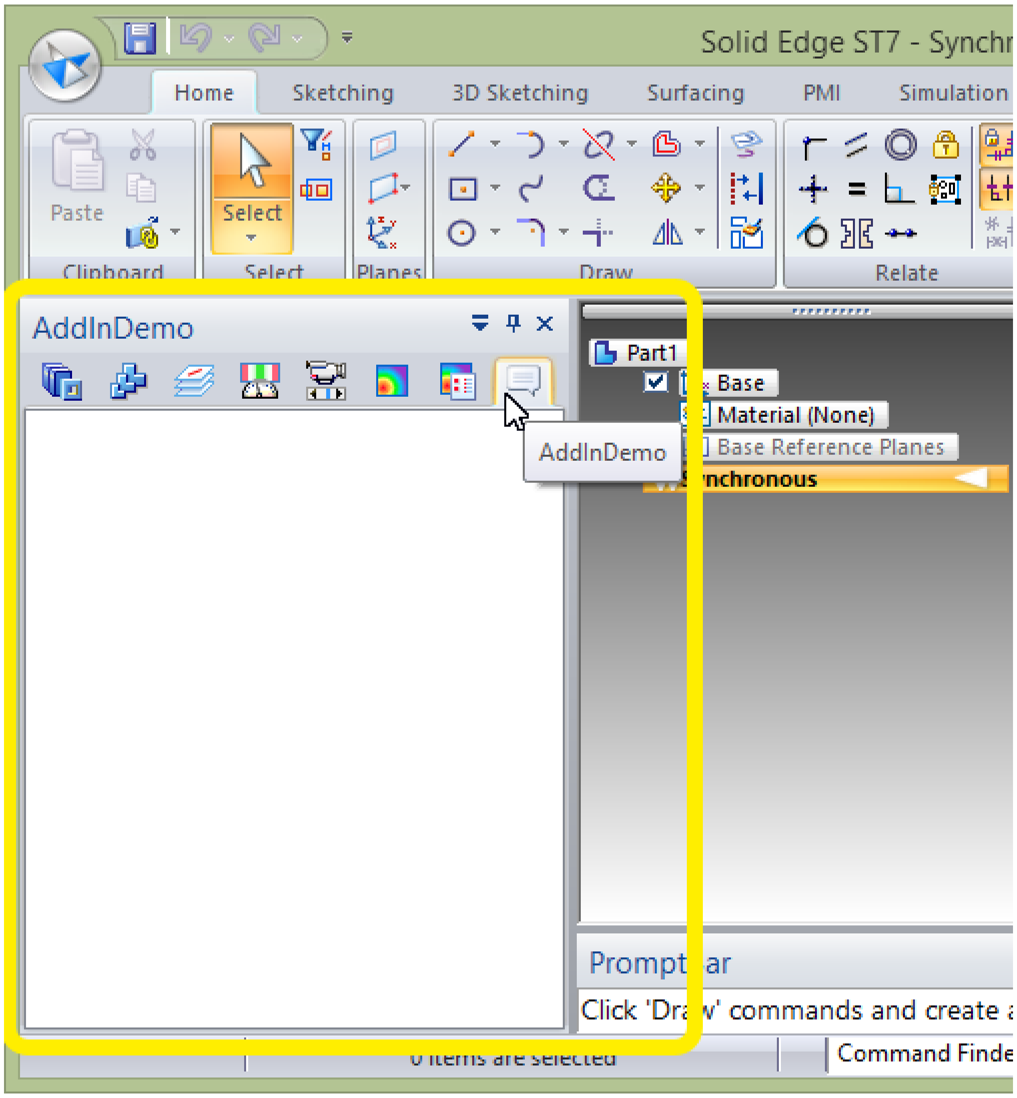
C++

```
// MyAddIn.cpp : Implementation of CMyAddIn

#include "stdafx.h"
#include "MyAddIn.h"

// CMyAddIn
#pragma region ISolidEdgeAddIn

STDMETHODIMP CMyAddIn::raw_OnConnection(IDispatch *pAppDispatch,
SolidEdgeFramework::SeConnectMode ConnectMode, SolidEdgeFramework::AddIn* pAddIn)
```

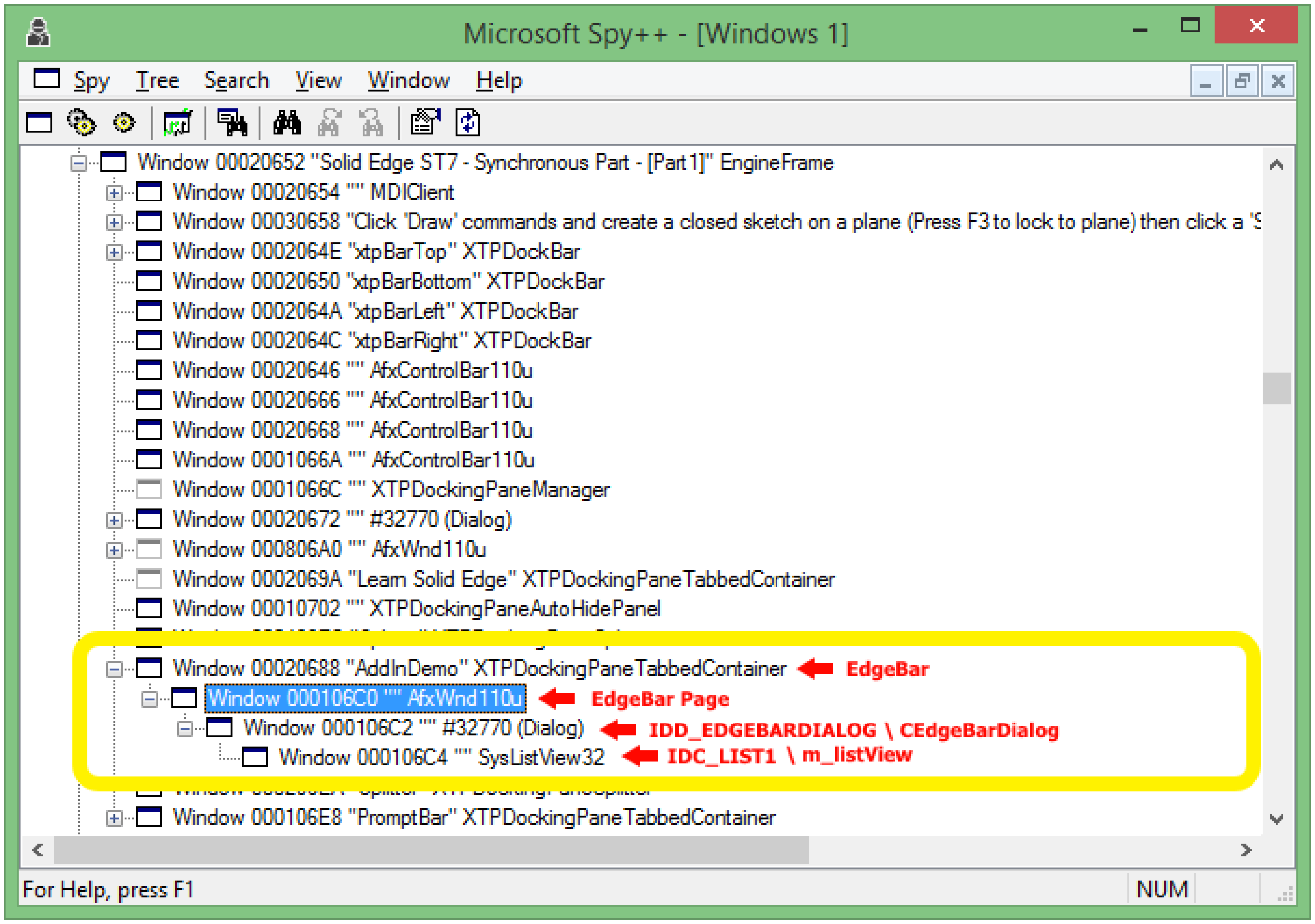
```
{
    HRESULT hr = S_OK;

    // Store smart pointer to addin. Note that the smart pointer is QueryInferface()'ing
for ISEAddInEx.
    m_pAddInEx = pAddIn;

    // Create instance of CEdgeBarController.
    hr = CEdgeBarControllerObj::CreateInstance(&m_pEdgeBarController);

    if (SUCCEEDED(hr))
    {
        // Manually AddRef() our local CEdgeBarController COM object.
        m_pEdgeBarController->AddRef();

        // Pass the ISEAddInEx pointer to the CEdgeBarController.
        hr = m_pEdgeBarController->SetAddInEx(m_pAddInEx);
    }

    return S_OK;
}

STDMETHODIMP CMyAddIn::raw_OnConnectToEnvironment(BSTR EnvironmentCatid, LPDISPATCH
pEnvironment, VARIANT_BOOL bFirstTime)
{
    return S_OK;
}

STDMETHODIMP CMyAddIn::raw_OnDisconnection(SolidEdgeFramework::SeDisconnectMode
DisconnectMode)
{
    HRESULT hr = S_OK;

    if (m_pEdgeBarController != NULL)
    {
        // Disconnect the CEdgeBarController by passing NULL.
        hr = m_pEdgeBarController->SetAddInEx(NULL);

        // Manually Release() our local CEdgeBarController COM object.
        m_pEdgeBarController->Release();

        m_pEdgeBarController = NULL;
    }

    return S_OK;
}

#pragma endregion
```

Build, Register and Debug

Now you're ready to Build, Register and Debug. Once everything compiles and you start debugging Solid Edge, you should have a new EdgeBar page.



To visualize the Window hierarchy, you can use Microsoft Spy++.


Microsoft Spy++ - [Windows 1]
Spy Tree Search View Window Help

```
Window 00020652 "Solid Edge ST7 - Synchronous Part - [Part1]" EngineFrame
  Window 00020654 "" MDIClient
  Window 00030658 "Click 'Draw' commands and create a closed sketch on a plane (Press F3 to lock to plane) then click a 'S
  Window 0002064E "xtpBarTop" XTPDockBar
  Window 00020650 "xtpBarBottom" XTPDockBar
  Window 0002064A "xtpBarLeft" XTPDockBar
  Window 0002064C "xtpBarRight" XTPDockBar
  Window 00020646 "" AfxControlBar110u
  Window 00020666 "" AfxControlBar110u
  Window 00020668 "" AfxControlBar110u
  Window 0001066A "" AfxControlBar110u
  Window 0001066C "" XTPDockingPaneManager
  Window 00020672 "" #32770 (Dialog)
  Window 000806A0 "" AfxWnd110u
  Window 0002069A "Learn Solid Edge" XTPDockingPaneTabbedContainer
  Window 00010702 "" XTPDockingPaneAutoHidePanel
  Window 00020688 "AddInDemo" XTPDockingPaneTabbedContainer   <- EdgeBar
    Window 000106C0 "" AfxWnd110u   <- EdgeBar Page
      Window 000106C2 "" #32770 (Dialog)   <- IDD_EDGEBARDIALOG \ CEdgeBarDialog
        Window 000106C4 "" SysListView32   <- IDC_LIST1 \ m_listView
  Window 000106E8 "PromptBar" XTPDockingPaneTabbedContainer
```

Conclusion

This article only focused on the bare minimum requirements to create an EdgeBar page in a Solid Edge AddIn but should serve as the foundation to create your own custom EdgeBar pages.

Resources

- Solid Edge ST7 SDK
- Solid Edge Developer Community
- Solid Edge Community on GitHub

License

Written By

Jason Newell
Software Developer (Senior) JasonNewell.NET
United States

Jason Newell is an Applications Architect for Ditch Witch and a Siemens PLM Solutions Partner specializing in custom development for Solid Edge. Jason also runs www.jasonnewell.net, a website dedicated to Solid Edge programming.

Comments and Discussions

5 messages have been posted for this article Visit **https://www.codeproject.com/Articles/844382/Solid-Edge-ST-AddIn-EdgeBar-CPP** to post and view comments on this article, or click **here** to get a print view with messages.

Permalink
Advertise
Privacy
Cookies
Terms of Use