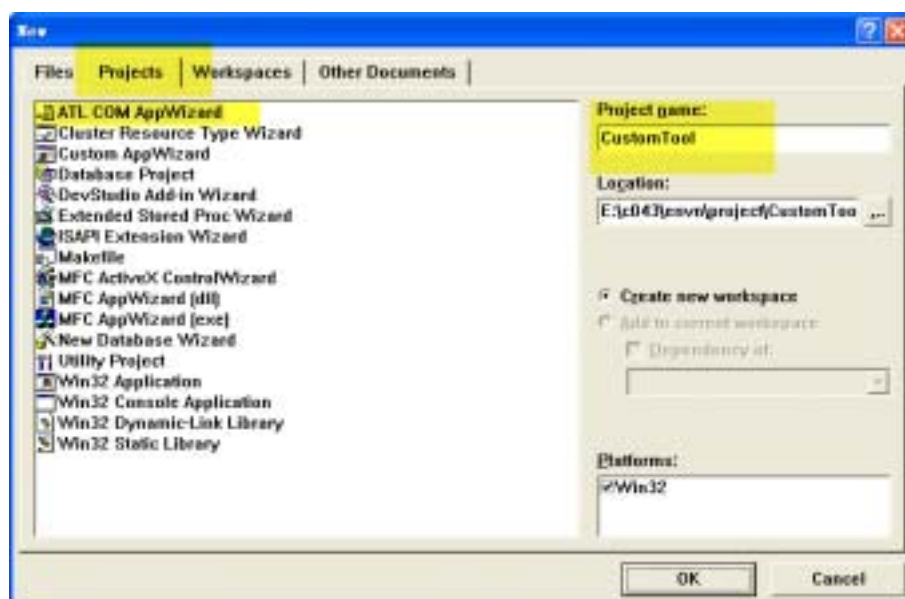


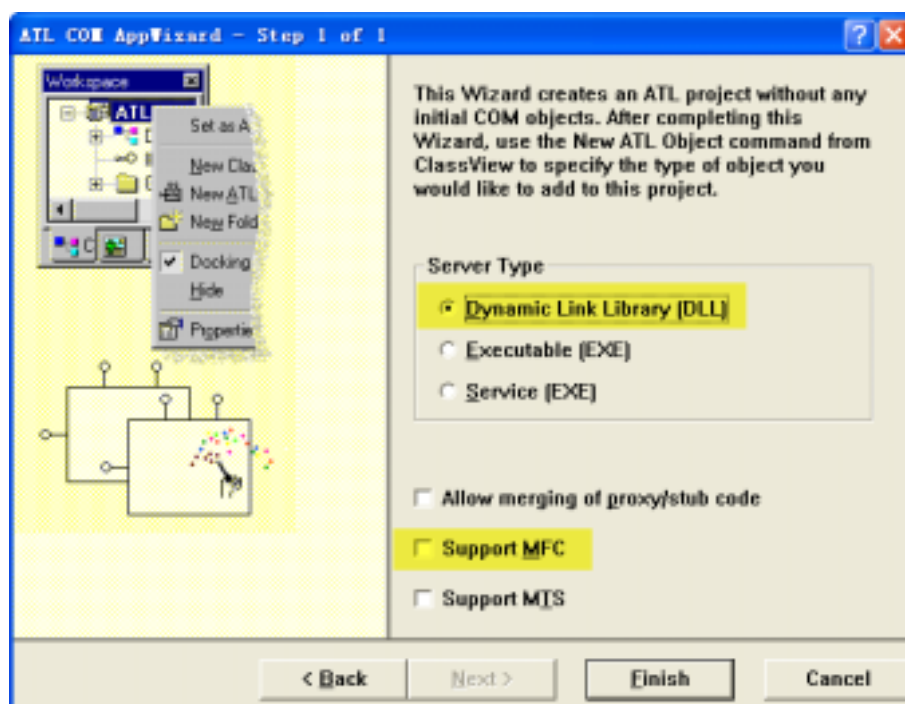
ArcObjects VC++开发插件教程

作者：yyi lyzbc

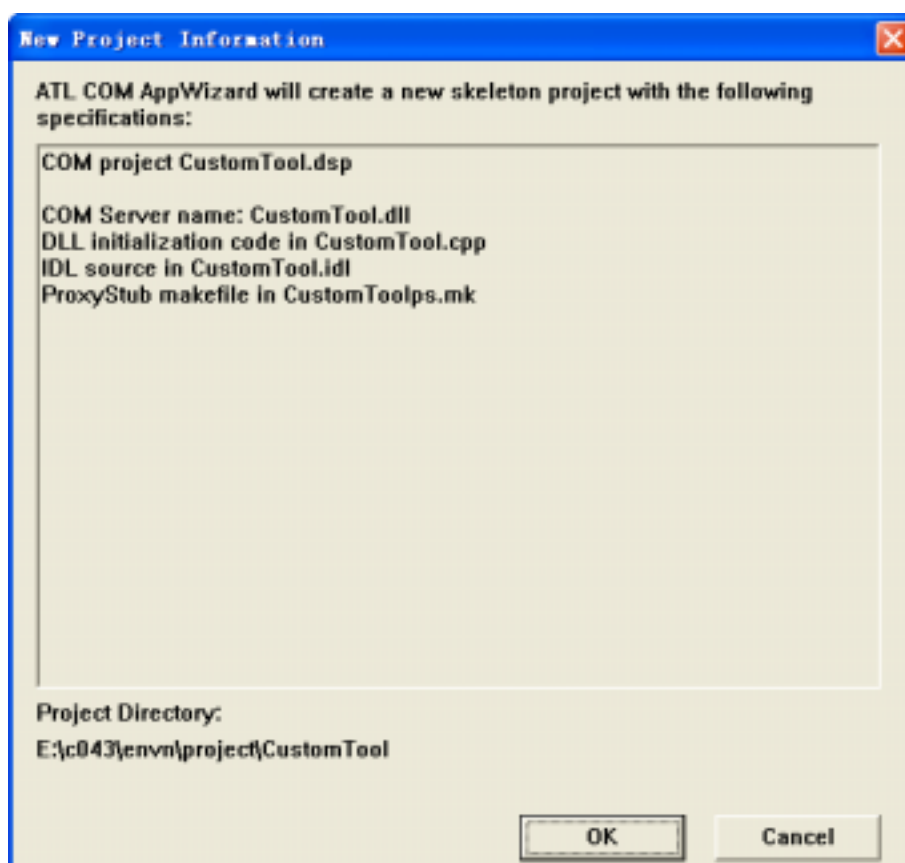
1 打开 vc6，file->new, 弹出 new 对话框，选择 projects 面板，选择 ATL COM AppWizard。
输入工程名 CustomTool。



2 点击 ok，选择默认设置即可。如果想在工程中使用 MFC, 则选中 Support MFC 选项。



3 点击 Finish。查看工程信息。

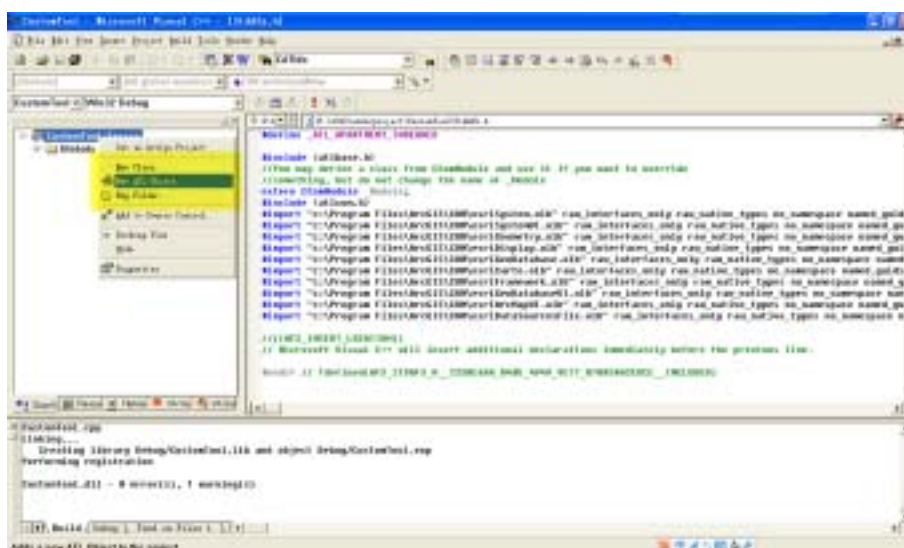


4 点击 ok。

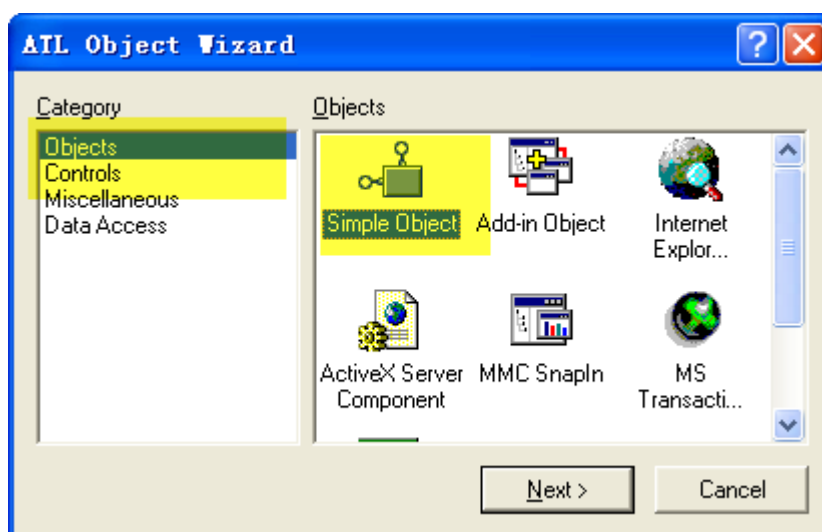
5 将下面代码拷贝到 Stdafx.h 文件中，放到 #include <atlcom.h>后面。这段代码的含义是引入 ao 的类型库，关于类型库的详细信息请参考 MSDN。

```
1.  #import "c:\Program Files\ArcGIS\COM\esriSystem.olb" raw_interfaces_only
    raw_native_types no_namespace named_guids exclude("OLE_COLOR", "OLE_HANDLE")
2.  #import "c:\Program Files\ArcGIS\COM\esriSystemUI.olb" raw_interfaces_only
    raw_native_types no_namespace named_guids exclude("OLE_COLOR", "OLE_HANDLE")
3.  #import "c:\Program Files\ArcGIS\COM\esriGeometry.olb" raw_interfaces_only
    raw_native_types no_namespace named_guids exclude("OLE_COLOR", "OLE_HANDLE")
    rename("wkbPoint","esri_wkbPoint")      rename("wkbPolygon","esri_wkbPolygon")
    rename("wkbMultiPoint","esri_wkbMultiPoint")
    rename("wkbMultiPolygon","esri_wkbMultiPolygon")
    rename("wkbGeometryCollection","esri_wkbGeometryCollection")
    rename("wkbXDR","esri_wkbXDR") rename("wkbNDR","esri_wkbNDR")
4.  #import "c:\Program Files\ArcGIS\COM\esriDisplay.olb" raw_interfaces_only
    raw_native_types no_namespace named_guids exclude("OLE_COLOR", "OLE_HANDLE")
5.  #import "c:\Program Files\ArcGIS\COM\esriGeoDatabase.olb" raw_interfaces_only
    raw_native_types no_namespace named_guids exclude("OLE_COLOR", "OLE_HANDLE")
6.  #import "c:\Program Files\ArcGIS\COM\esriCarto.olb" raw_interfaces_only
    raw_native_types no_namespace named_guids exclude("OLE_COLOR", "OLE_HANDLE")
7.  #import "c:\Program Files\ArcGIS\COM\esriFramework.olb" raw_interfaces_only
    raw_native_types no_namespace named_guids exclude("OLE_COLOR", "OLE_HANDLE")
8.  #import "c:\Program Files\ArcGIS\COM\esriGeoDatabaseUI.olb"
    raw_interfaces_only raw_native_types no_namespace named_guids
    exclude("OLE_COLOR", "OLE_HANDLE")
9.  #import "c:\Program Files\ArcGIS\COM\esriArcMapUI.olb" raw_interfaces_only
    raw_native_types no_namespace named_guids exclude("OLE_COLOR", "OLE_HANDLE")
10. #import "c:\Program Files\ArcGIS\COM\esriDataSourcesFile.olb"
    raw_interfaces_only raw_native_types no_namespace named_guids
    exclude("OLE_COLOR", "OLE_HANDLE")
```

6 切换到 class View，选中 CustomTool 工程，右键。选择 new ATL Object。



7 弹出 new ATL Object wizard。选择 simple Object。



8 单击 next，在 name 面板输入 name：CustomTool bar

9 切换到 Attribute 面板，Threading Model 选择 Single，Interface 选择 Custom，Aggregation 选择 no。并勾选上 Support ISupportErrorInfo。



10 点击确定。

11 切换到 FileView，双击 CustomTool.idl，打开 CustomTool.idl 文件，在

```
importlib("stdole32.tlb");
```

```
importlib("stdole2.tlb");
```

后面加入下面代码：

```
1. importlib("c:\Program Files\ArcGIS\COM\esriSystemUI.olb");
2. importlib("c:\Program Files\ArcGIS\COM\esriFramework.olb");
```

在 [default] interface ICustomToolBar;

下面加入下面代码：

```
interface IToolBarDef;
```

使 CustomToolBar 支持 IToolBarDef 接口。

12 双击 CustomToolBar.h 文件，打开 CustomToolBar.h 文件。使 CCustomToolBar 类继承 IToolBarDef 接口，即在

public ICustomToolBar 后面添加如下代码：

, public IToolBarDef 记得要用逗号分隔开。

在宏 BEGIN_COM_MAP(CCustomToolBar)和 END_COM_MAP()中间添加以下代码：

```
COM_INTERFACE_ENTRY(IToolBarDef)
```

将下面函数添加为类的成员函数，保证为 public 的：

```
1. // IToolBarDef
2. STDMETHOD(get_ItemCount)(LONG* numItems);
```

```
3.  STDMETHODCALLTYPE(GetItemInfo)(LONG pos, IItemDef* itemDef);
4.  STDMETHODCALLTYPE(get_Name)(BSTR* Name);
5.  STDMETHODCALLTYPE(get_Caption)(BSTR* Name);
```

上面的代码是 IToolBarDef 的接口函数。要在 CustomToolBar 中实现。

13 双击 CustomToolBar.cpp, 打开 CustomToolBar.cpp 文件，将下面代码拷贝到 CustomToolBar.cpp 文件中。

其中 get_ItemCount 函数为得到当前工具条工具的个数，GetItemInfo 得到工具条中第 pos 个工具的信息，

其中 CustomTool.MyTool 为我们将要实现工具的 ProgID。

```
1.  // IToolBarDef
2.  STDMETHODCALLTYPE CCustomToolBar::get_ItemCount(LONG* numItems)
3.  {
4.  if (0 == numItems)
5.      return E_POINTER;
6.
7.  // Set how many commands will be on the toolbar
8.  *numItems = 1;
9.
10. return S_OK;
11. }
12. STDMETHODCALLTYPE CCustomToolBar::GetItemInfo(LONG pos, IItemDef* itemDef)
13. {
14. if (0 == itemDef)
15.     return E_POINTER;
16.
17. // Define the commands that will be on the toolbar. The 1st command
18. // will be the custom command MyCustomTool. The 2nd and 3rd commands will
19. // be the builtin AddData commands and ZoomIn tool.
20. // ID is the ProgID of the command. Group determines whether the command
21. // begins a new group on the toolbar
```

```
22.
23. HRESULT hr;
24. switch (pos)
25. {
26.     case 0:
27.         hr = itemDef->put_ID(CComBSTR(L"CustomTool.MyTool"));
28.         if (FAILED(hr)) return hr;
29.
30.         hr = itemDef->put_Group(VARIANT_FALSE);
31.         if (FAILED(hr)) return hr;
32.         break;
33.     default:
34.         return E_INVALIDARG;
35. }
36.
37. return S_OK;
38. }
39. STDMETHODCALLTYPE CCustomToolbar::get_Name(BSTR* Name)
40. {
41.     if (0 == Name)
42.         return E_POINTER;
43.
44.     // Set the internal name of the toolbar.
45.     *Name = ::SysAllocString(L"CustomTool");
46.
47.     return S_OK;
48. }
49. STDMETHODCALLTYPE CCustomToolbar::get_Caption(BSTR* Name)
50. {
51.     if (0 == Name)
52.         return E_POINTER;
```

```
53.  
54. // Set the string that appears as the toolbar's title  
55. *Name = ::SysAllocString(L"CustomTool");  
56.  
57. return S_OK;  
58. }
```

14 使用上面的方法再创建一个 ATL Object，选项同 CustomTool bar。命名为 MyTool。

15 重新打开 customTool.idl，在[default] interface IMyTool;后面加上如下代码：

interface ICommand;

interface ITool;

和 CustomTool bar 支持 IToolBarDef 接口类似，给 CMyTool 类添加 interface ICommand;interface ITool;的支持。

步骤同上，首先 CMyTool 继承 ICommand，ITool。

在宏 BEGIN_COM_MAP(CMyTool)和 END_COM_MAP()中间添加对 ICommand，ITool 的支持：

COM_INTERFACE_ENTRY(ICommand)

COM_INTERFACE_ENTRY(ITool)

定义 ICommand，ITool 的接口函数：

将下列代码加到 CMyTool 类中作为 public 的成员函数。

```
1. STDMETHODCALLTYPE(get_Enabled)(VARIANT_BOOL* Enabled);  
2. STDMETHODCALLTYPE(get_Checked)(VARIANT_BOOL* Checked);  
3. STDMETHODCALLTYPE(get_Name)(BSTR* Name);  
4. STDMETHODCALLTYPE(get_Caption)(BSTR* Caption);  
5. STDMETHODCALLTYPE(get_Tooltip)(BSTR* Tooltip);  
6. STDMETHODCALLTYPE(get_Message)(BSTR* Message);  
7. STDMETHODCALLTYPE(get_HelpFile)(BSTR* HelpFile);  
8. STDMETHODCALLTYPE(get_HelpContextID)(LONG* helpID);  
9. STDMETHODCALLTYPE(get_Bitmap)(OLE_HANDLE* Bitmap);  
10. STDMETHODCALLTYPE(get_Category)(BSTR* categoryName);  
11. STDMETHODCALLTYPE(OnCreate)(IDispatch* hook);  
12. STDMETHODCALLTYPE(OnClick)();  
13. // ITool
```

```
14. STDMETHOD(get_Cursor)(OLE_HANDLE* Cursor);
15. STDMETHOD(OnMouseDown)(LONG button, LONG shift, LONG x, LONG y);
16. STDMETHOD(OnMouseMove)(LONG button, LONG shift, LONG x, LONG y);
17. STDMETHOD(OnMouseUp)(LONG button, LONG shift, LONG x, LONG y);
18. STDMETHOD(OnDbClick)();
19. STDMETHOD(OnKeyDown)(LONG keyCode, LONG shift);
20. STDMETHOD(OnKeyUp)(LONG keyCode, LONG shift);
21. STDMETHOD(OnContextMenu)(LONG x, LONG y, VARIANT_BOOL* handled);
22. STDMETHOD(Refresh)(OLE_HANDLE hdc);
23. STDMETHOD(Deactivate)(VARIANT_BOOL* complete);
```

将下列代码加到 CMyTool 类中作为 private 变量。

```
1. HBITMAP m_hBitmap;
2. HCURSOR m_hCursor;
3. IApplicationPtr m_ipApp;
4. IDocumentPtr m_ipDoc;
```

修改 CShowXY 的构造函数：

将

CMyTool ()

```
{
}
```

替换成：

```
    CMyTool ();
~CMyTool ();
```

在 MyTool.cpp 文件中实现上面的函数，拷贝下面代码到 MyTool.cpp 中。

```
1. // CMyTool
2. CMyTool::CMyTool()
3. {
4.     m_hBitmap = :oadBitmap(_Module.m_hInst, MAKEINTRESOURCE(IDB_BITMAP1));
5. }
```

```
6.  CMyTool::~CMyTool()
7.  {
8.      :eleteObject(m_hBitmap);
9.  }
10. // ICommand
11. STDMETHODIMP CMyTool::get_Enabled(VARIANT_BOOL* Enabled)
12. {
13.     if (0 == Enabled)
14.         return E_POINTER;
15.     // Add some logic here to specify when the command should
16.     // be enabled. In this example, the command is always enabled.
17.     *Enabled = VARIANT_TRUE;
18.     return S_OK;
19. }
20. STDMETHODIMP CMyTool::get_Checked(VARIANT_BOOL* Checked)
21. {
22.     if (0 == Checked)
23.         return E_POINTER;
24.     return E_NOTIMPL;
25. }
26. STDMETHODIMP CMyTool::get_Name(BSTR* Name)
27. {
28.     if (0 == Name)
29.         return E_POINTER;
30.     // Set the internal name of this command. By convention, this
31.     // name string contains the category and caption of the command.
32.     *Name = ::SysAllocString(L"DeveloperSamples_MyTool");
33.     return S_OK;
34. }
35. STDMETHODIMP CMyTool::get_Caption(BSTR* Caption)
36. {
```

```
37.  if (0 == Caption)
38.      return E_POINTER;
39.  // Set the string that appears when the command is used as a
40.  // menu item.
41.  *Caption = ::SysAllocString(L"MyTool");
42.  return S_OK;
43. }
44. STDMETHODIMP CMyTool::get_Tooltip(BSTR* Tooltip)
45. {
46.  if (0 == Tooltip)
47.      return E_POINTER;
48.  // Set the string that appears in the screen tip.
49.  *Tooltip = ::SysAllocString(L"MyTool");
50.  return S_OK;
51. }
52. STDMETHODIMP CMyTool::get_Message(BSTR* Message)
53. {
54.  if (0 == Message)
55.      return E_POINTER;
56.  // Set the message string that appears in the statusbar of the
57.  // application when the mouse passes over the command.
58.  *Message = ::SysAllocString(L"This is my custom tool");
59.  return S_OK;
60. }
61. STDMETHODIMP CMyTool::get_HelpFile(BSTR* HelpFile)
62. {
63.  if (0 == HelpFile)
64.      return E_POINTER;
65.  return E_NOTIMPL;
66. }
67. STDMETHODIMP CMyTool::get_HelpContextID(LONG* helpID)
```

```
68. {
69.     if (0 == helpID)
70.         return E_POINTER;
71.     return E_NOTIMPL;
72. }
73. STDMETHODCALLTYPE CMyTool::get_Bitmap(OLE_HANDLE* Bitmap)
74. {
75.     if (0 == Bitmap)
76.         return E_POINTER;
77.     // Set the bitmap of the command. The m_hBitmap variable is set
78.     // in class constructor
79.     *Bitmap = (OLE_HANDLE)m_hBitmap;
80.     return S_OK;
81. }
82. STDMETHODCALLTYPE CMyTool::get_Category(BSTR* categoryName)
83. {
84.     if (0 == categoryName)
85.         return E_POINTER;
86.     // Set the category of this command. This determines where the
87.     // command appears in the Commands panel of the Customize dialog.
88.     *categoryName = ::SysAllocString(L"Developer Samples");
89.     return S_OK;
90. }
91. STDMETHODCALLTYPE CMyTool::OnCreate(IDispatch* hook)
92. {
93.     // The hook argument is a pointer to Application object.
94.     // Establish a hook to the application
95.     m_ipApp = hook;
96.     m_ipApp->get_Document(&m_ipDoc);
97.     return S_OK;
98. }
```

```
99. STDMETHODCALLTYPE CMyTool::OnClick()
100. {
101.     // Add some code to do some action when the command is clicked. In this
102.     // example, a message box is displayed.
103.     CComBSTR bstrCaption(L"MyTool Command");
104.     CComBSTR bstrMsg(L"Clicked on my command.");
105.     USES_CONVERSION;
106.     ::MessageBox(0, OLE2T(bstrMsg), OLE2T(bstrCaption), MB_OK);
107.     return S_OK;
108. }
109. // ITool
110. STDMETHODCALLTYPE CMyTool::get_Cursor(OLE_HANDLE* Cursor)
111. {
112.     if (0 == Cursor)
113.         return E_POINTER;
114.     // Set the cursor of the command. The m_pCursor variable is set
115.     // in Class_Initialize
116.     *Cursor = (OLE_HANDLE)m_hCursor;
117.     return S_OK;
118. }
119. STDMETHODCALLTYPE CMyTool::OnMouseDown(LONG button, LONG shift, LONG x, LONG y)
120. {
121.     // Add some code to do some action when the mouse button is pressed.
122.     // This example displays the X and Y coordinates of the
123.     // left mouse button click in the statusbar message in ArcMap.
124.     // Button, X, and Y are passed in as arguments to this sub procedure.
125.     // Check to see if left button is pressed
126.     if (1 == button)
127.     {
128.         IMxApplicationPtr ipMxApp;
129.         ipMxApp = m_ipApp;
```

```
130.     HRESULT hr;
131.     IAppDisplayPtr ipAppDisplay;
132.     ipMxApp->get_Display(&ipAppDisplay);
133.     // Convert x and y to map units. m_ipApp is set in OnCreate.
134.     IDisplayTransformationPtr ipDisplayTrans;
135.     ipAppDisplay->get_DisplayTransformation(&ipDisplayTrans);
136.     IPointPtr ipPoint;
137.     hr = ipDisplayTrans->ToMapPoint(x, y, &ipPoint);
138.     // Set the statusbar message.
139.     double dX, dY;
140.     TCHAR tX[16], tY[16];
141.
142.     ipPoint->get_X(&dX);
143.     ipPoint->get_Y(&dY);
144.     _sntprintf(tX, 16, _T("%5.3lf"), dX);
145.     _sntprintf(tY, 16, _T("%5.3lf"), dY);
146.     CComBSTR bstrMsg;
147.     bstrMsg = tX;
148.     bstrMsg += L", ";
149.     bstrMsg += tY;
150.     IStatusBarPtr ipStatusBar;
151.     m_ipApp->get_StatusBar(&ipStatusBar);
152.     ipStatusBar->put_Message(0, bstrMsg);
153. }
154. return S_OK;
155. }
156. STDMETHODIMP CMyTool::OnMouseMove(LONG button, LONG shift, LONG x, LONG y)
157. {
158.     // Add some code to do some action when the mouse is moved.
159.     // This example changes the statusbar message.
160.     IStatusBarPtr ipStatusBar;
```

```
161. m_ipApp->get_StatusBar(&ipStatusBar);
162. ipStatusBar->put_Message(0, CComBSTR(L"ITool_OnMouseMove"));
163. return S_OK;
164. }
165. STDMETHODIMP CMyTool::OnMouseUp(LONG button, LONG shift, LONG x, LONG y)
166. {
167.     return E_NOTIMPL;
168. }
169. STDMETHODIMP CMyTool::OnDbClick()
170. {
171.     // Add some code to do some action on double-click.
172.     // This example makes the builtin Select Graphics Tool the active tool.
173.
174.     ICommandBarsPtr ipCommandBars;
175.     m_ipDoc->get_CommandBars(&ipCommandBars);
176.     // The identifier for the Select Graphics Tool
177.     CComVariant vVariant("{C22579D1-BC17-11D0-8667-0000F8751720}");
178.     ICommandItemPtr ipSelectTool;
179.     ipCommandBars->Find(vVariant, VARIANT_FALSE, VARIANT_FALSE, &ipSelectTool);
180.     // Set the current tool of the application to be the Select Graphics Tool
181.     m_ipApp->putref_CurrentTool(ipSelectTool);
182.     return S_OK;
183. }
184. STDMETHODIMP CMyTool::OnKeyDown(LONG keyCode, LONG shift)
185. {
186.     // Add some code to do some action when a keyboard button is pressed.
187.     // This example changes the statusbar message.
188.     IStatusBarPtr ipStatusBar;
189.     m_ipApp->get_StatusBar(&ipStatusBar);
190.     ipStatusBar->put_Message(0, CComBSTR(L"ITool_OnKeyDown"));
191.     return S_OK;
```

```
192. }
193. STDMETHODIMP CMyTool::OnKeyUp(LONG keyCode, LONG shift)
194. {
195.     // Add some code to do some action when a keyboard button is released.
196.     // This example changes the statusbar message.
197.     IStatusBarPtr ipStatusBar;
198.     m_ipApp->get_StatusBar(&ipStatusBar);
199.     ipStatusBar->put_Message(0, CComBSTR(L"ITool_OnKeyUp"));
200.     return S_OK;
201. }
202. STDMETHODIMP CMyTool::OnContextMenu(LONG x, LONG y, VARIANT_BOOL* handled)
203. {
204.     if (0 == handled)
205.         return E_POINTER;
206.     // Add some code to show a custom context menu when there is a right click.
207.     // This example creates a new context menu with one macro item
208.     ICommandBarsPtr ipCmdBars;
209.     m_ipDoc->get_CommandBars(&ipCmdBars);
210.     // Create a new context menu
211.     ICommandBarPtr ipShortCut;
212.     ipCmdBars->Create(CComBSTR(L"MyShortCut"),          esriCmdBarTypeShortcutMenu,
        &ipShortCut);
213.     // Add an item to it
214.     ICommandItemPtr ipItem;
215.     CComVariant vFaceID(4);
216.     USHORT uAction(0);
217.     CComVariant vIndex(0);
218.     ipShortCut->CreateMacroItem(CComBSTR(L"MyMacro"),    &vFaceID,    &uAction,
        &vIndex, &ipItem);
219.     // Display the menu
220.     ICommandItemPtr ipChoice;
```

```
221.  ipShortCut->Popup(0, 0, &ipChoice);
222.  // Let the application know that you handled the OnContextMenu event.
223.  // If you don't do this, the standard context menu will be displayed after
224.  // this custom context menu.
225.  *handled = VARIANT_TRUE;
226.  return S_OK;
227. }
228. STDMETHODCALLTYPE CMyTool::Refresh(OLE_HANDLE hdc)
229. {
230.     return S_OK;
231. }
232. STDMETHODCALLTYPE CMyTool::eactivate(VARIANT_BOOL* complete)
233. {
234.     if (0 == complete)
235.         return E_POINTER;
236.     // Deactivate the tool. If set to False (the default), you cannot interact
237.     // with any other tools because this tool cannot be interrupted by another
238.     // tool.
239.     *complete = VARIANT_TRUE;
240.     return S_OK;
241. }
```

从 hook 得到 m_ipApp，从 m_ipApp 得到 m_ipDoc。

得到 m_ipDoc 以后就可以对地图文档进行操作了。

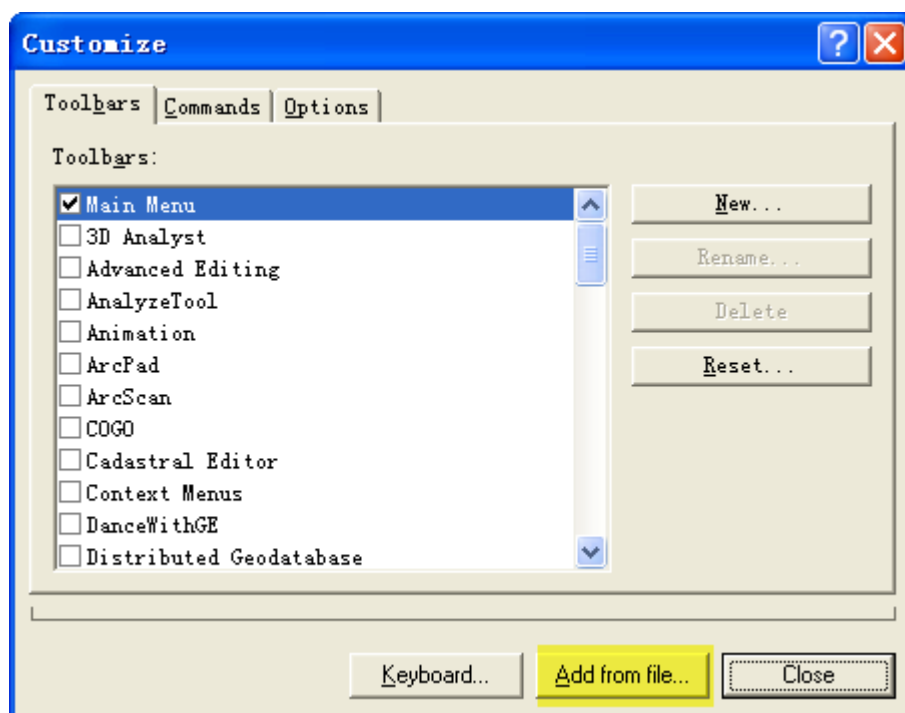
其他函数都比较好理解，主要是对鼠标事件的响应已经一些状态信息的获取。

注意：要导入一个 bitmap 的资源文件做为工具图标，默认 ID 为 IDB_BITMAP1。

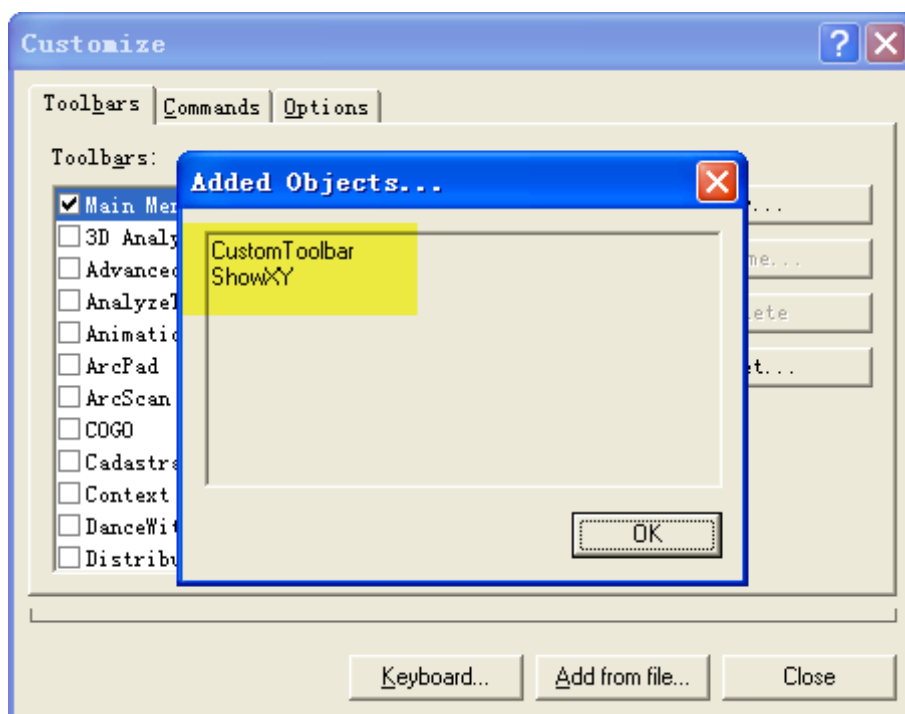
16 编译工程。

17 打开 ArcMap 右键在工具栏空白处单击，选择 customize...

18 选择 add from file...

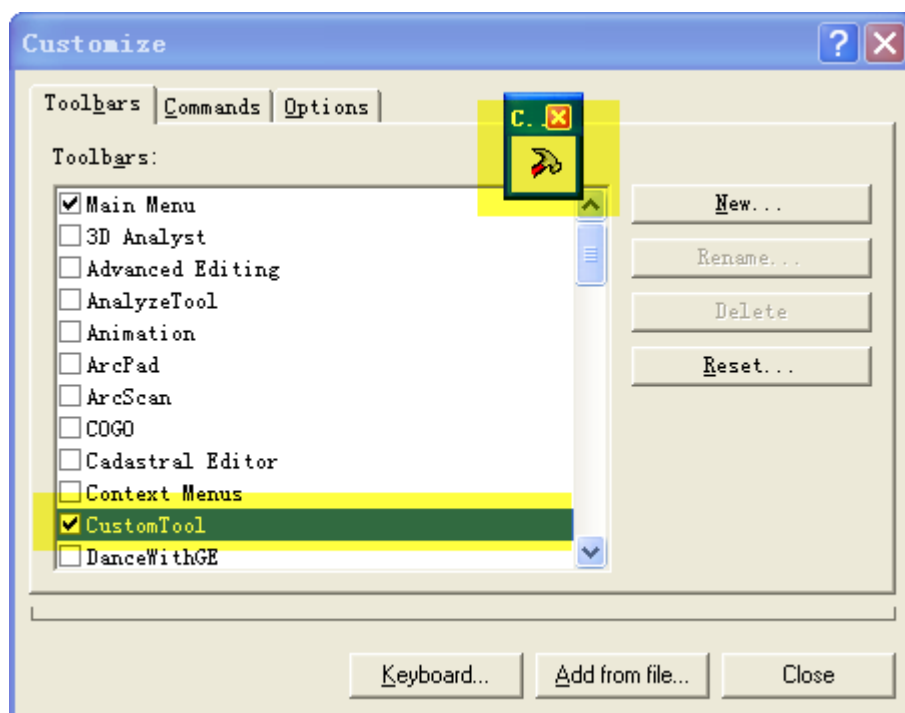


19 定位到 debug 目录，选择 CustomTool.dll



20 点击打开

21 点击 ok，选择 CustomTool



22 点击 close。