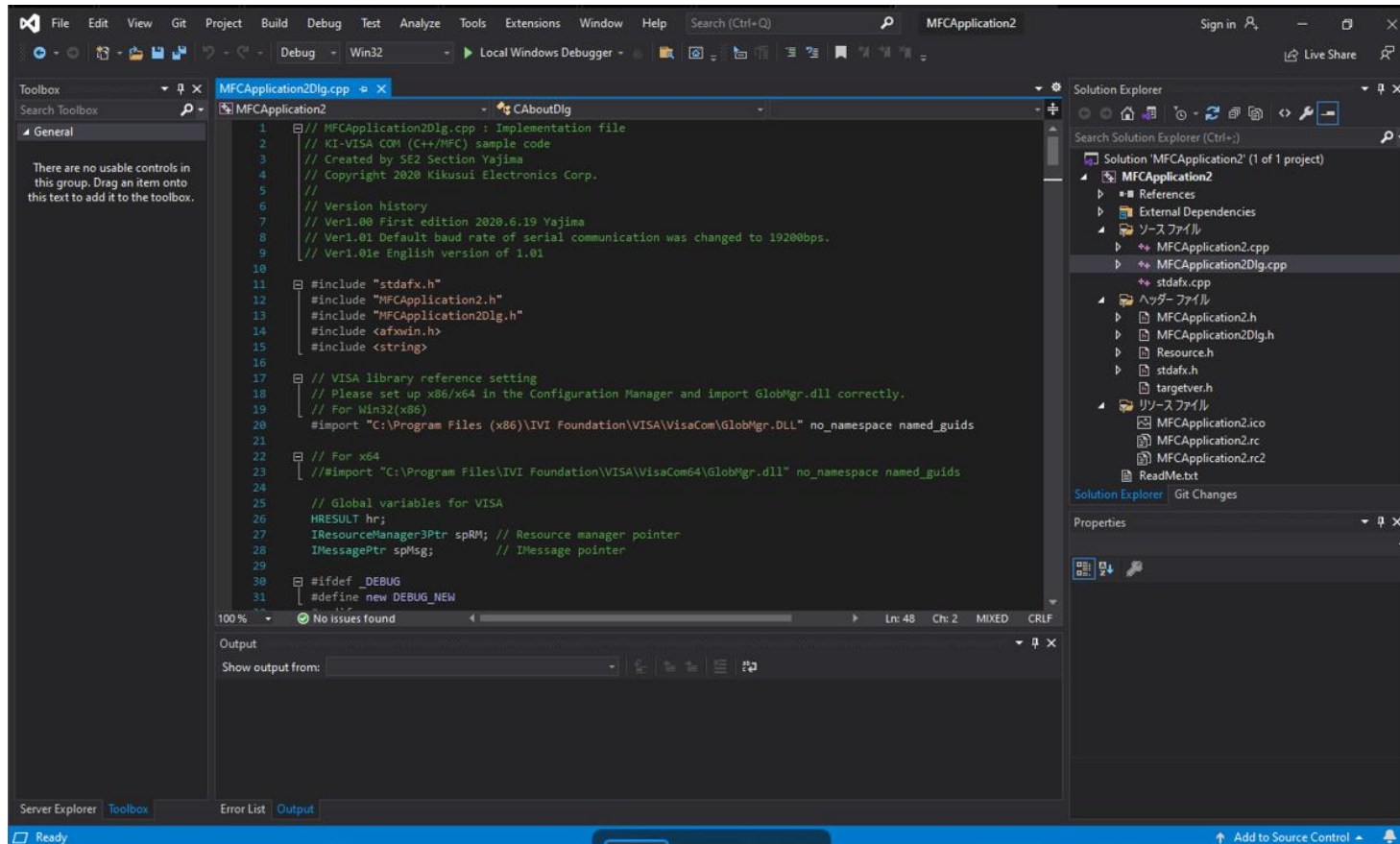


Introduction to sample code for device control using KI-VISA library (for C++/MFC)



Contents

- Introduction to our sample code
- What can our sample code do?
- What is needed to run our sample code?
- Details of our sample code
- Form and Event Handler
- How to manage the reference
- How to solve a build error
- Guidebook / Contact Us

Introduction to our sample code

- **Purpose of our sample code**

Our sample code simplifies the control of measurement instrumentation by utilizing the KI-VISA library for Microsoft Visual C++/MFC.

You can easily control devices, such as power supplies, measuring instruments and electronic loads, by modifying the initial value of our sample code, according to the communication interface manuals.

As a typical example, we provide our sample code specifically for our DC power supply PMX series.

- **Intended user**

Our sample code is designed for development engineers who need guidance in creating programming codes to control the devices using the KI-VISA library.

What can our sample code do?

- **Connect a PC and device**

Select the VISA resource name from the FindResource pull-down menu. Click **Open** to open the VISA session and establish a communication link with the device.

- **Send a command**

[Command] button

Select the command from the Command pull-down menu. Click **Command** to send the command to the device.

- **Send a query command**

[Query] button

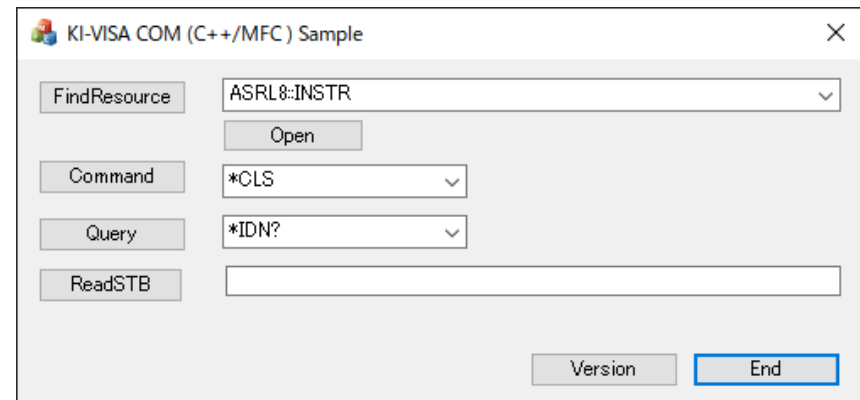
Select the command from the Query pull-down menu. Click **Query** to send the query command to the device. The result data is returned and displayed in the text box.

- **Read a status byte register**

[ReadSTB] button

Click **ReadSTB** to execute the ReadSTB method. The value of the status byte register is displayed in the text box.

*Available only when connected via the GPIB or USB interface.



What is needed to run our sample code?

- **Windows PC with the following requirements**

- KI-VISA library (Download is free of charge.)

<https://global.kikusui.co.jp/dri-fir-upd/ki-visa/>

*Please note that if another VISA library, such as NI-VISA or Keysight VISA, is already installed, this library is not necessary. The reference setting is required, please see [How to manage the reference.](#)

- Microsoft Visual Studio 2013 or later

*The Visual Studio Express edition does not support Visual C++/MFC, but our sample code can run on the Visual Studio Community edition (Please check the license conditions from the link below):

<https://visualstudio.microsoft.com/vs/community/>

- **Connection cable**

You will need one of the following connection cables.

- USB connection: USB Cable (USB Type A-Type B)
- Serial connection: Serial cable (9-pin cross type)
USB-RS232C conversion cable
(e.g., REX-USB60F manufactured by RATOC Systems, Inc.)
*If your PC has a serial port, the conversion cable is not required.
- GPIB connection: USB-GPIB conversion cable
(Example: USB-GPIB-HS manufactured by National Instruments)
*Use the cable that supports the VISA COM library.
You cannot use the GPIB-USB cable by RATOC Systems, Inc.
- LAN connection: LAN cable (cross-type)
*In case of AUTO-MDIX LAN interface, a straight cable is also available.

- **Device such as measuring instrument or power supply**

You will need the device with the following interfaces:

- GPIB interface
- Serial interface (RS232C)
- USB interface (USBTMC)
- LAN interface (VXI-11, HiSLIP)

Details of our sample code

- **Contents of the ZIP file**

- The set of project including solution (.sln) file

- **Programming Language**

Visual C++/MFC 2013

* We have verified that the file can be loaded into Visual Studio 2019 Community edition with the retargeted project.

- **Code Examples**

- Finding the VISA resource name of the device (FindResource method)
- How to open/close the VISA session (Open/Close method)
- Setting RS-232C baud rate
- Sending a command string (WriteString method)
- Receiving a query message (ReadString method)
- Reading the status byte by serial polling (ReadSTB method)

- **Precautions before implementing our sample code**

- You can modify our sample code, but please note that they are provided only as a reference of your programming.
- This sample code does not guarantee complete operation in all environments.
- Error handling has been minimized to simplify our sample code.
We strongly recommend handling all errors (try-catch) on your actual system.

Form and Event Handler

KI-VISA COM (C++/MFC) Sample

FindResource ASRL8:INSTR Open

Command *CLS

Query *IDN?

ReadSTB

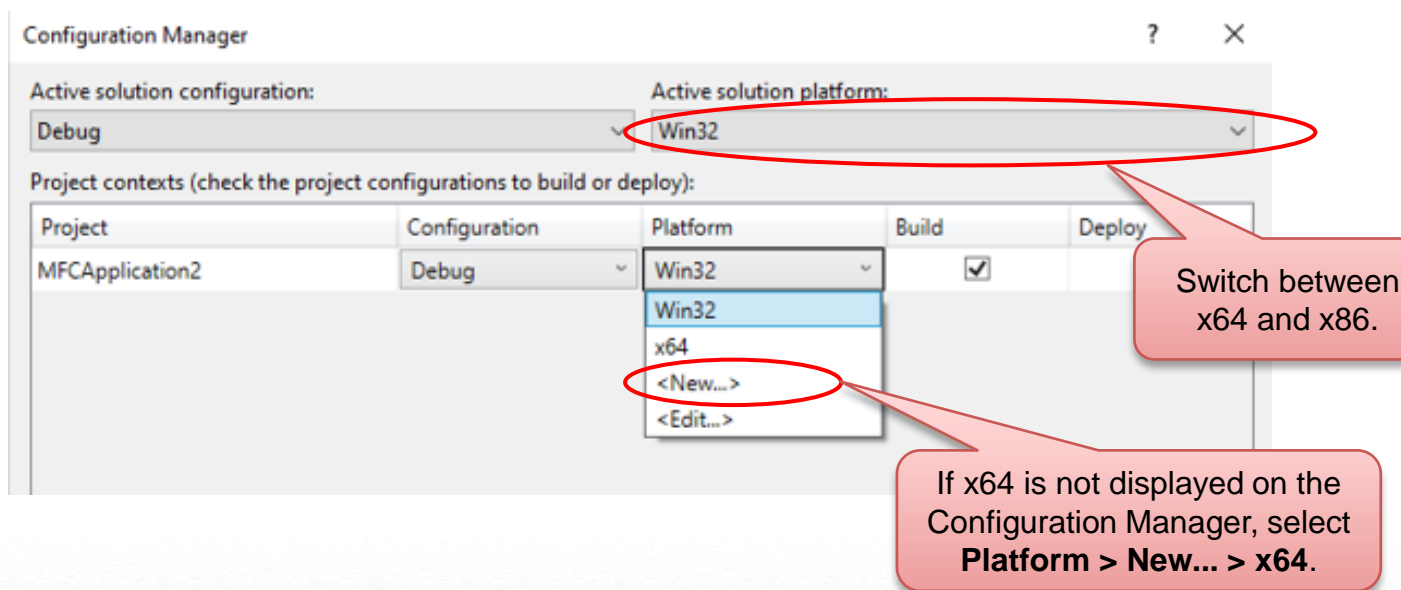
Version End

```
219 //-----  
220 // FindResource Button Handler  
221 //-----  
222 void CMFCApplication2Dlg::OnBnClickedButton4() { ... }  
259 //-----  
260 // Open button handler  
261 //-----  
262 void CMFCApplication2Dlg::OnBnClickedButton7() { ... }  
327 //-----  
328 // Command button handler  
329 //-----  
330 void CMFCApplication2Dlg::OnBnClickedButton1() { ... }  
359 //-----  
360 // Query button handler  
361 //-----  
362 void CMFCApplication2Dlg::OnBnClickedButton2() { ... }  
398 //-----  
399 // ReadSTB button handler  
400 //-----  
401 void CMFCApplication2Dlg::OnBnClickedButton3() { ... }  
430 //-----  
431 // End button handler  
432 //-----  
433 void CMFCApplication2Dlg::OnBnClickedCancel() { ... }  
459 //-----  
460 // Version button event handler  
461 //-----  
462 void CMFCApplication2Dlg::OnBnClickedButton5() { ... }  
468
```

How to manage the reference

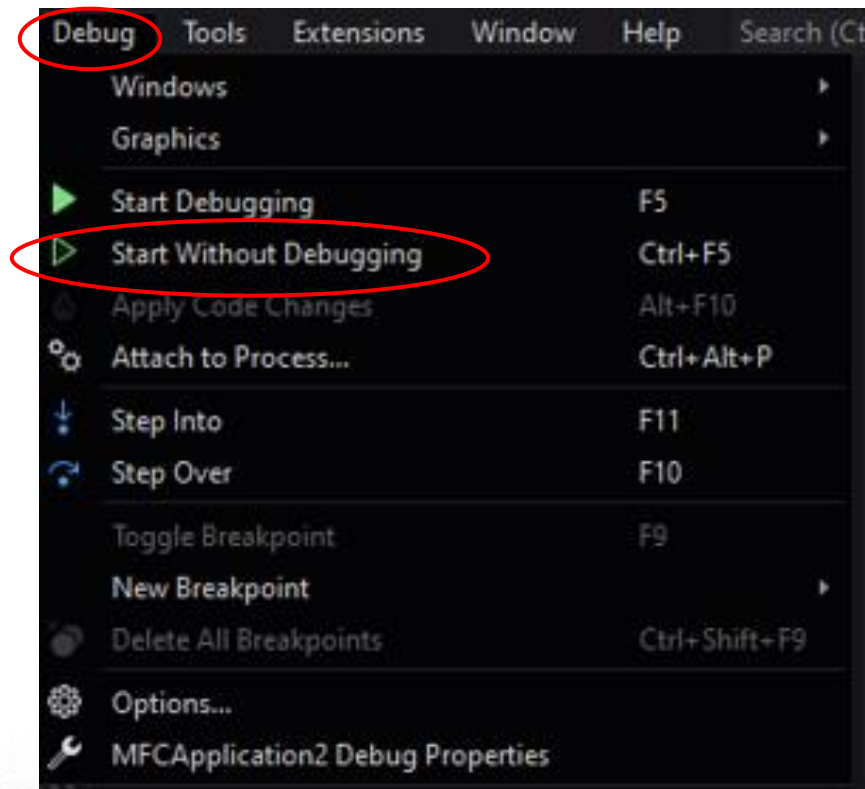
When creating a new project or changing the bitness of the application, you can comment out or in the code by following these steps below. By default, the library's 32-bit version is referenced.

```
17 // VISA library reference setting
18 // Please set up x86/x64 in the Configuration Manager and import GlobMgr.dll correctly.
19 // For Win32(x86)
20 #import "C:\Program Files (x86)\IVI Foundation\Visa\VisaCom\GlobMgr.DLL" no_namespace named_guids
21
22 // For x64
23 // #import "C:\Program Files\IVI Foundation\Visa\VisaCom64\GlobMgr.dll" no_namespace named_guids
24
```



How to solve a build error - Part 1/2

- When you first build the program with our sample code, it may result in a build error. However, once you perform the action "Debug > Start Without Debugging", the debug file will be generated. You can then build the program properly.



How to solve a build error - Part 2/2

Build Error:

The following errors may occur when building our sample code:

E1696: Cannot open source file "afxwin.h"

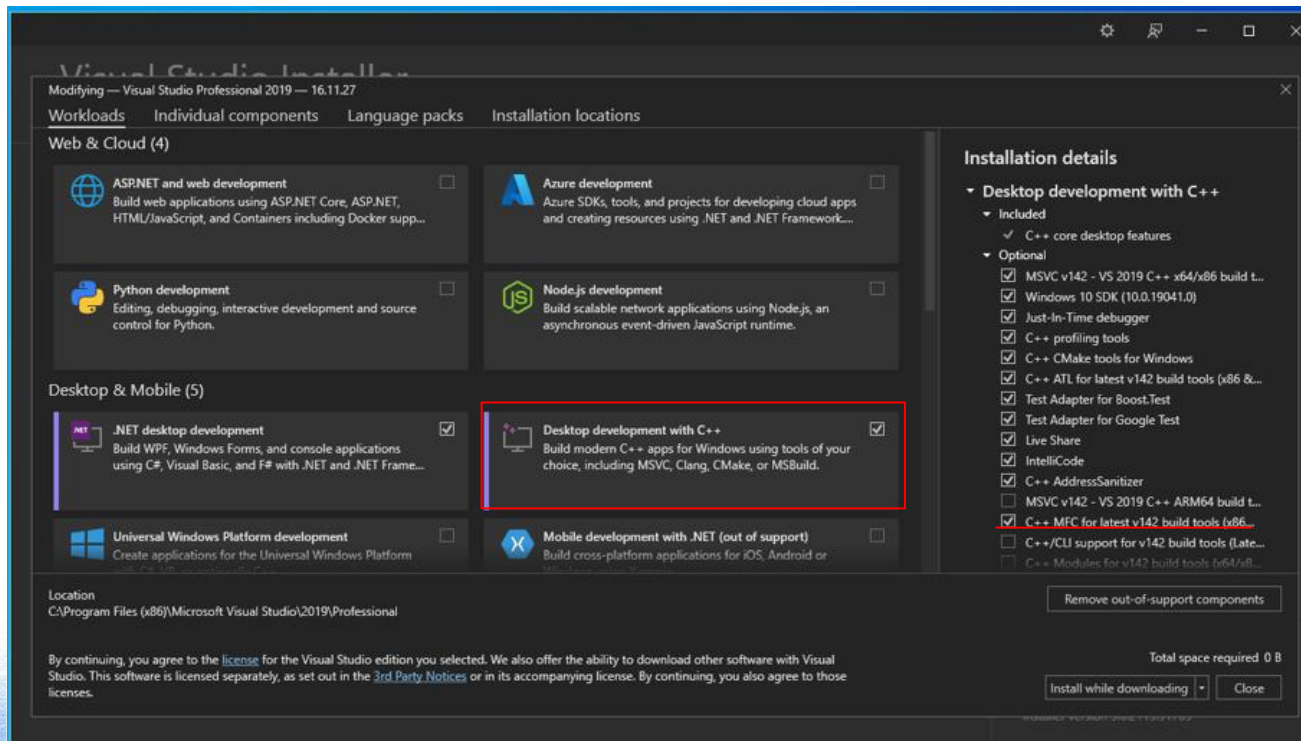
C1083: Cannot open include file "afxwin.h": No such file or directory

Solution:

To resolve this issue, follow these steps to install the MFC build tool.

How to install the MFC build tool:

1. Open Visual Studio and go to **Tools > Get Tools and Features...** to open the Visual Studio installer.
2. Check **"Desktop development with C++"** and **"C++ MFC for latest v142 build tools (x86 & x64)"**. Click **Modify** to initiate the installation process.



Guidebook / Contact Us

- Please download the KI-VISA programming guidebook for reference (Note: The guidebook is available in Japanese text only):

<https://kikusui.co.jp/kikusupport/downloads/guidebook/programing-guidebook/>

- For any inquiries regarding our sample code, please contact us using the following inquiry form:

<https://global.kikusui.co.jp/contact/lead/>

Please include the phrase “To the person in charge of the sample code” in the text and provide your question.

```

1 // MFCApplication2Dlg.cpp : Implementation file
2 // KI-VISA COM (C++/MFC) sample code
3 // Created by SE2 Section Yajima
4 // Copyright 2020 Kikusui Electronics Corp.
5 //
6 // Version history
7 // Ver1.00 First edition 2020.6.19 Yajima
8 // Ver1.01 Default baud rate of serial communication was changed to 19200bps.
9 // Ver1.01e English version of 1.01
10
11 #include "stdafx.h"
12 #include "MFCApplication2.h"
13 #include "MFCApplication2Dlg.h"
14 #include <afxwin.h>
15 #include <string>
16
17 // VISA library reference setting
18 // Please set up x86/x64 in the Configuration Manager and import GlobMgr.dll correctly.
19 // For Win32(x86)
20 #import "C:\Program Files (x86)\IVI Foundation\VISA\VisaCom\GlobMgr.DLL" no_namespace named_guids
21
22 // For x64
23 #import "C:\Program Files\IVI Foundation\VISA\VisaCom64\GlobMgr.dll" no_namespace named_guids
24
25 // Global variables for VISA
26 HRESULT hr;
27 IResourceManager3Ptr spRM; // Resource manager pointer
28 IMessagePtr spMsg;        // IMessage pointer
29
30 #ifdef _DEBUG
31 #define new DEBUG_NEW
32 #endif
33
34 // CAboutDlg dialog used for application version information
35 class CAboutDlg : public CDialogEx
36 {
37 public:
38     CAboutDlg();
39
40     // Dialogue data
41     enum { IDD = IDD_ABOUTBOX };
42
43 protected:
44     virtual void DoDataExchange(CDataExchange* pDX); // DDX/DDV support
45
46
47
48     // Implementation
49 protected:
50     DECLARE_MESSAGE_MAP()
51 };
52
53
54 CAboutDlg::CAboutDlg() : CDialogEx(CAboutDlg::IDD)
55 {
56
57 }
58
59 void CAboutDlg::DoDataExchange(CDataExchange* pDX)
60 {
61     CDialogEx::DoDataExchange(pDX);
62 }
63
64 BEGIN_MESSAGE_MAP(CAboutDlg, CDialogEx)
65 END_MESSAGE_MAP()
66
67
68 // CMFCApplication2Dlg dialog.
69 CMFCApplication2Dlg::CMFCApplication2Dlg(CWnd* pParent /*=NULL*/)
70 : CDialogEx(CMFCApplication2Dlg::IDD, pParent)
71 {
72     m_hIcon = AfxGetApp()->LoadIcon(IDR_MAINFRAME);

```

```

73 }
74
75 void CMFCApplication2Dlg::DoDataExchange(CDataExchange* pDX)
76 {
77     CDialogEx::DoDataExchange(pDX);
78     DDX_Control(pDX, IDC_COMBO1, m_combo1);
79     DDX_Control(pDX, IDC_Command, m_Command);
80     DDX_Control(pDX, IDC_Query, m_Query);
81     DDX_Control(pDX, IDC_Result, c_Result);
82 }
83
84 BEGIN_MESSAGE_MAP(CMFCApplication2Dlg, CDialogEx)
85     ON_WM_SYSCOMMAND()
86     ON_WM_PAINT()
87     ON_WM_QUERYDRAGICON()
88     ON_BN_CLICKED(IDOK, &CMFCApplication2Dlg::OnBnClickedOk)
89     ON_BN_CLICKED(IDC_BUTTON1, &CMFCApplication2Dlg::OnBnClickedButton1)
90     ON_BN_CLICKED(IDC_BUTTON2, &CMFCApplication2Dlg::OnBnClickedButton2)
91     ON_BN_CLICKED(IDC_BUTTON3, &CMFCApplication2Dlg::OnBnClickedButton3)
92     ON_BN_CLICKED(IDC_BUTTON4, &CMFCApplication2Dlg::OnBnClickedButton4)
93     ON_BN_CLICKED(IDCANCEL, &CMFCApplication2Dlg::OnBnClickedCancel)
94     ON_BN_CLICKED(IDC_BUTTON7, &CMFCApplication2Dlg::OnBnClickedButton7)
95     ON_BN_CLICKED(IDC_BUTTON5, &CMFCApplication2Dlg::OnBnClickedButton5)
96 END_MESSAGE_MAP()
97
98 // -----
99 // CMFCApplication2Dlg message handler
100 // (VISA resource manager creation is done here)
101 // -----
102 BOOL CMFCApplication2Dlg::OnInitDialog()
103 {
104     CDialogEx::OnInitDialog();
105
106     // "About..." Add the menu to the system menu.
107
108     // IDM_ABOUTBOX must be within the range of system commands.
109     ASSERT((IDM_ABOUTBOX & 0xFFF0) == IDM_ABOUTBOX);
110     ASSERT(IDM_ABOUTBOX < 0xF000);
111
112     CMenu* pSysMenu = GetSystemMenu(FALSE);
113     if (pSysMenu != NULL)
114     {
115         BOOL bNameValid;
116         CString strAboutMenu;
117         bNameValid = strAboutMenu.LoadString(IDS_ABOUTBOX);
118         ASSERT(bNameValid);
119         if (!strAboutMenu.IsEmpty())
120         {
121             pSysMenu->AppendMenu(MF_SEPARATOR);
122             pSysMenu->AppendMenu(MF_STRING, IDM_ABOUTBOX, strAboutMenu);
123         }
124     }
125
126     // Set the icon for this dialog. If your application's main window is not a dialog,
127     // the Framework automatically sets this setting for you.
128     SetIcon(m_hIcon, TRUE); // Set the large icon
129     SetIcon(m_hIcon, FALSE); // Set the small icon
130
131     //-----
132     // initialization
133     //-----
134
135     // Create a resource manager
136     hr = CoInitialize(NULL); // COM initialization
137     hr = spRM.CreateInstance(CLSID_ResourceManager); // Create a resource manager
138
139     // FindResource execution
140     OnBnClickedButton4(); // Call the event handler of the FindResource button
141
142     // Add a command string to the combo box
143     m_Command.AddString(_T("VOLT 3.3"));
144     m_Command.AddString(_T("VOLT 5.0"));

```



```

145 m_Command.AddString(_T("VOLT 18.0"));
146 m_Command.AddString(_T("OUTP 1"));
147 m_Command.AddString(_T("OUTP 0"));
148 m_Command.AddString(_T("*CLS"));
149 m_Command.AddString(_T("*RST"));
150 m_Command.SetCurSel(0); // Display the first item.
151
152 // Add a query string to the combo box
153 m_Query.AddString(_T("*IDN?"));
154 m_Query.AddString(_T("MEAS:VOLT?"));
155 m_Query.AddString(_T("MEAS:CURRE?"));
156 m_Query.AddString(_T("OUTP?"));
157 m_Query.AddString(_T("SYST:ERR?"));
158 m_Query.AddString(_T("STAT:OPER:COND?"));
159
160 m_Query.SetCurSel(0); // Display the first item.
161
162 return TRUE; // TRUE, except when the focus is set to control.
163 }
164
165 void CMFCApplication2Dlg::OnSysCommand(UINT nID, LPARAM lParam)
166 {
167     if ((nID & 0xFFF0) == IDM_ABOUTBOX)
168     {
169         CAboutDlg dlgAbout;
170         dlgAbout.DoModal();
171     }
172     else
173     {
174     }
175 }
176
177
178 // If you want to add a minimize button to a dialog,
179 // you need the code below to draw the icon.
180 // For MFC applications that use a document/view model,
181 // this will be set automatically by the Framework.
182
183 void CMFCApplication2Dlg::OnPaint()
184 {
185     if (IsIconic())
186     {
187         CPaintDC dc(this); // device context for drawing
188
189         SendMessage(WM_ICONERASEBKGND, reinterpret_cast<WPARAM>(dc.GetSafeHdc()), 0);
190
191         // The center of the client's rectangular area
192         int cxIcon = GetSystemMetrics(SM_CXICON);
193         int cyIcon = GetSystemMetrics(SM_CYICON);
194         CRect rect;
195         GetClientRect(&rect);
196         int x = (rect.Width() - cxIcon + 1) / 2;
197         int y = (rect.Height() - cyIcon + 1) / 2;
198
199         // Draw the icon.
200         dc.DrawIcon(x, y, m_hIcon);
201     }
202     else
203     {
204         CDialogEx::OnPaint();
205     }
206 }
207
208 // The system calls this function to get the cursor to display
209 // when the user is dragging the minimized window.
210 HCURSOR CMFCApplication2Dlg::OnQueryDragIcon()
211 {
212     return static_cast<HCURSOR>(m_hIcon);
213 }
214
215 // OK button handler
216 void CMFCApplication2Dlg::OnBnClickedOk()
217 {

```

```

217 CDialogEx::OnOK();
218 }
219 //-----
220 // FindResource Button Handler
221 //-----
222 void CMFCApplication2Dlg::OnBnClickedButton4()
223 {
224     // Search for valid visa resource strings
225     SAFEARRAY* pSA = NULL;
226     try {
227         // FindResource Execution
228         pSA = spRM->FindRsrc(L"?*INSTR");
229     }
230     catch (...) {
231         // If no VISA resource is found
232         AfxMessageBox(_T("No VISA resources were found"));
233     }
234
235     if (pSA) {
236         BSTR* rgElems = NULL;
237         ::SafeArrayAccessData(pSA, (PVOID*)&rgElems);
238         ASSERT(rgElems);
239
240         LONG lLBound, lUBound;
241         ::SafeArrayGetLBound(pSA, 1, &lLBound);
242         ::SafeArrayGetUBound(pSA, 1, &lUBound);
243
244         //Clear the value of the combo box.
245         m_combo1.ResetContent();
246
247         //Add a resource name to the combo box.
248         for (long lNdx = lLBound; lNdx <= lUBound; lNdx++) {
249             m_combo1.AddString(rgElems[lNdx]);
250         }
251
252         m_combo1.SetCurSel(0); // Select the first item in the combo box
253
254         ::SafeArrayUnaccessData(pSA);
255         ::SafeArrayDestroyData(pSA);
256     }
257 }
258 //-----
259 // Open button handler
260 //-----
261 void CMFCApplication2Dlg::OnBnClickedButton7()
262 {
263     CString sResourceName; // Resource name
264     bstr_t bResourceName; // Resource name (bstr type)
265
266     // COM initialization process
267     hr = CoInitialize(NULL);
268
269     // Get resource name text
270     m_combo1.GetWindowText(sResourceName); // Get the resource name (text) displayed in the combo
271
272     bResourceName = sResourceName.AllocSysString(); // Resource name variable is converted to CString -> R
273     by AllocSysString() and passed.
274
275     try
276     {
277         // VISA Resource Open
278         spMsg = spRM->Open(bResourceName, NO_LOCK, 0, L "");
279
280         // When writing it as a direct value,
281         // add "L" like "spMsg = spRM->Open(L"ASRL8::INSTR", NO_LOCK, 0, L "");
282
283         // Setting for each interface
284         IGpibPtr gpib;
285         ISerialPtr seri;
286         ITcpipInstrPtr tcp;
287         IUsbPtr usb;

```

```

288
289     switch (spMsg->HardwareInterfaceType)
290     {
291         case 1: //GPiB
292             gpib = (IGpibPtr)spMsg;
293             gpib->RepeatAddressingEnabled = true;
294             gpib->UnaddressingEnabled = false;
295
296             break;
297
298         case 4: //ASRL (Serial)
299             seri = (ISerialPtr)spMsg;
300             seri->BaudRate = 19200; //Baud rate
301             seri->DataBits = 8; //Data bit
302             seri->StopBits = SerialStopBits(ASRL_STOP_ONE); //Stop bit (1 bit)
303             seri->Parity = SerialParity(ASRL_PAR_NONE); //No parity
304             seri->FlowControl = SerialFlowControl(ASRL_FLOW_NONE); //No flow control
305             break;
306
307         case 6: //LAN
308             tcp = (ITcpipInstrPtr)spMsg;
309
310             break;
311
312         case 7: //USB
313             usb = (IUsbPtr)spMsg;
314
315             break;
316
317         default:
318
319             break;
320     }
321     AfxMessageBox(sResourceName + _T("is Opened"));
322
323 }
324
325 catch (...)
326 {
327     AfxMessageBox(_T("An error occurred on opening."));
328 }
329 }
330 //-----
331 // Command button handler
332 //-----
333 void CMFCApplication2Dlg::OnBnClickedButton1()
334 {
335     // Command transmission sample
336
337     int r;
338     CString sCommand;
339     bstr_t bCommand;
340
341     // Getting the command string
342     m_Command.GetWindowText(sCommand); // Get a command string from a combo box
343     sCommand += "\n"; // Add <LF> delimiter
344     bCommand = sCommand.AllocSysString(); // Convert to bstr type
345
346     try
347     {
348         //Command transmission
349         r = spMsg->WriteString(bCommand);
350
351         // If you want to use direct values,
352         // add "L" like " r = spMsg->WriteString(L "OUTP 1¥n");".
353     }
354     catch (...)
355     {
356         AfxMessageBox(_T("An error occurred on 'Command'."));
357     }
358 }
359 }

```

```

360 //-----
361 // Query button handler
362 //-----
363 void CMFCApplication2Dlg::OnBnClickedButton2()
364 {
365     //Sample query reception
366     int r;
367     std::string rcvData;
368     CString sQuery;
369     bstr_t bQuery;
370
371     //Get query string
372     m_Query.GetWindowText(sQuery);    // Get a query string from a combo box
373     sQuery += "\n";                  // Add <LF> delimiter
374     bQuery = sQuery.AllocSysString(); // Convert to bstr type
375
376     try
377     {
378         // query command transmission
379         r = spMsg->WriteString(bQuery);
380
381         // If you want to describe it as a direct value,
382         // add "L" like "r = spMsg->WriteString(L "*IDN?*IDN?\n");"
383
384         // Receive result message
385         rcvData = spMsg->ReadString(1024);
386
387         // Display the dialog box
388         CA2T msg(rcvData.c_str());
389         c_Result.SetWindowText(msg);
390
391     }
392     catch (...)
393     {
394         AfxMessageBox(_T("An error has occurred in 'Query'."));
395     }
396 }
397
398 //-----
399 // ReadSTB button handler
400 //-----
401 void CMFCApplication2Dlg::OnBnClickedButton3()
402 {
403     // ReadSTB sample
404
405     short stb;
406     CString str;
407
408     // Check the HardwareInterfaceType as ReadSTB is not available on the serial interface
409     if (spMsg->HardwareInterfaceType!=4)
410     {
411         try
412         {
413             stb = spMsg->ReadSTB();
414             str.Format(_T("%d"), stb);
415             c_Result.SetWindowText(str);
416         }
417         catch (...)
418         {
419             AfxMessageBox(_T("An error occurred on 'ReadSTB'."));
420         }
421     }
422     else
423     {
424         AfxMessageBox(_T("ReadSTB is not available on the serial interface"));
425     }
426 }
427
428 //-----
429 // End button handler

```

```

432 //-----
433 void CMFCApplication2Dlg::OnBnClickedCancel()
434 {
435     //Close VISA resources and close the dialog.
436     if (spMsg != nullptr)
437     {
438         AfxMessageBox(_T("Close VISA Resources' and exit. "));
439         try
440         {
441             spMsg->Close();
442             CoUninitialize(); // Destroy the COM object
443             CDialogEx::OnCancel();
444         }
445         catch (...)
446         {
447             AfxMessageBox(_T("An error occurred when closing. "));
448             CDialogEx::OnCancel();
449         }
450     }
451     else
452     {
453         AfxMessageBox(_T("End. "));
454         CDialogEx::OnCancel();
455     }
456 }
457 //-----
458 // Version button event handler
459 //-----
460 void CMFCApplication2Dlg::OnBnClickedButton5()
461 {
462     CAboutDlg dlgAbout;
463     dlgAbout.DoModal();
464 }
465

```