China-pub.com
下载

第一部分　程序员必读

第1章　对程序错误的处理

在开始介绍Microsoft Windows 的特性之前，必须首先了解Windows的各个函数是如何进行错误处理的。

当调用一个Windows函数时，它首先要检验传递给它的的各个参数的有效性，然后再设法执行任务。如果传递了一个无效参数，或者由于某种原因无法执行这项操作，那么操作系统就会返回一个值，指明该函数在某种程度上运行失败了。表 1-1列出了大多数Windows函数使用的返回值的数据类型。

表1-1　Windows函数常用的返回值类型

数 据 类 型	表示失败的值
VOID	该函数的运行不可能失败。Windows函数的返回值类型很少是VOID
BOOL	如果函数运行失败，那么返回值是0，否则返回的是非0值。最好对返回值进行测试，以确定它是0还是非0。不要测试返回值是否为TRUE
HANDLE	如果函数运行失败，则返回值通常是NULL，否则返回值为HANDLE，用于标识你可以操作的一个对象。注意，有些函数会返回一个句柄值INVALID_ HANDLE_VALUE，它被定义为-1。函数的Platform SDK文档将会清楚地说明该函数运行失败时返回的是NULL还是INVALID_HANDLE_VALID
PVOID	如果函数运行失败，则返回值是NULL，否则返回PVOID，以标识数据块的内存地址
LONG/DWORD	这是个难以处理的值。返回数量的函数通常返回 LONG或DWORD。如果由于某种原因，函数无法对想要进行计数的对象进行计数，那么该函数通常返回 0或-1（根据函数而定）。如果调用的函数返回了LONG/DWORD，那么请认真阅读Platform SDK文档，以确保能正确检查潜在的错误

一个Windows函数返回的错误代码对了解该函数为什么会运行失败常常很有用。Microsoft公司编译了一个所有可能的错误代码的列表，并且为每个错误代码分配了一个32位的号码。

从系统内部来讲，当一个Windows函数检测到一个错误时，它会使用一个称为线程本地存储器(thread-local storage)的机制，将相应的错误代码号码与调用的线程关联起来(线程本地存储器将在第21章中介绍)。这将使线程能够互相独立地运行，而不会影响各自的错误代码。当函数返回时，它的返回值就能指明一个错误已经发生。若要确定这是个什么错误，请调用GetLastError函数：

```
DWORD GetLastError();
```

该函数只返回线程的32位错误代码。

当你拥有32位错误代码的号码时，必须将该号码转换成更有用的某种对象。WinError.h头文件包含了Microsoft公司定义的错误代码的列表。下面显示了该列表的某些内容，使你能够看到它的大概样子：

```
// MessageId: ERROR_SUCCESS
//
// MessageText:
```

China-pub.com 下载

```
//
//  The operation completed successfully.
//
#define ERROR_SUCCESS                    0L

#define NO_ERROR 0L                                   // dderror
#define SEC_E_OK                         ((HRESULT)0x00000000L)

//
// MessageId: ERROR_INVALID_FUNCTION
//
// MessageText:
//
//  Incorrect function.
//
#define ERROR_INVALID_FUNCTION           1L    // dderror

//
// MessageId: ERROR_FILE_NOT_FOUND
//
// MessageText:
//
//  The system cannot find the file specified.
//
#define ERROR_FILE_NOT_FOUND             2L

//
// MessageId: ERROR_PATH_NOT_FOUND
//
// MessageText:
//
//  The system cannot find the path specified.
//
#define ERROR_PATH_NOT_FOUND             3L

//
// MessageId: ERROR_TOO_MANY_OPEN_FILES
//
// MessageText:
//
//  The system cannot open the file.
//
#define ERROR_TOO_MANY_OPEN_FILES        4L

//
// MessageId: ERROR_ACCESS_DENIED
//
// MessageText:
//
//  Access is denied.
//
#define ERROR_ACCESS_DENIED              5L
```

如你所见，每个错误都有3种表示法：一个消息ID（这是你可以在源代码中使用的一个宏，以便与GetLastError的返回值进行比较），消息文本（对错误的英文描述）和一个号码（应该避

China-pub.com
下载

免使用这个号码，可使用消息ID）。请记住，这里只显示了WinError.h头文件中的很少一部分内容，整个文件的长度超过21000行。

当Windows函数运行失败时，应该立即调用GetLastError函数。如果调用另一个Windows函数，它的值很可能被改写。

注意 GetLastError能返回线程产生的最后一个错误。如果该线程调用的Windows函数运行成功，那么最后一个错误代码就不被改写，并且不指明运行成功。有少数Windows函数并不遵循这一规则，它会更改最后的错误代码；但是Platform SDK文档通常指明，当函数运行成功时，该函数会更改最后的错误代码。

Windows 98 许多Windows 98的函数实际上是用Microsoft公司的16位Windows 3.1产品产生的16位代码来实现的。这种比较老的代码并不通过GetLastError之类的函数来报告错误，而且Microsoft公司并没有在Windows 98中修改16位代码，以支持这种错误处理方式。对于我们来说，这意味着Windows 98中的许多Win32函数在运行失败时不能设置最后的错误代码。该函数将返回一个值，指明运行失败，这样你就能够发现该函数确实已经运行失败，但是你无法确定运行失败的原因。

有些Windows函数之所以能够成功运行，其中有许多原因。例如，创建指明的事件内核对象之所以能够取得成功，是因为你实际上创建了该对象，或者因为已经存在带有相同名字的事件内核对象。你应搞清楚成功的原因。为了将该信息返回，Microsoft公司选择使用最后错误代码机制。这样，当某些函数运行成功时，就能够通过调用GetLadtError函数来确定其他的一些信息。对于具有这种行为特性的函数来说，Platform SDK文档清楚地说明了GetLastError函数可以这样使用。请参见该文档，找出CreateEvent函数的例子。

进行调试的时候，监控线程的最后错误代码是非常有用的。在Microsoft Visual studio 6.0中，Microsoft的调试程序支持一个非常有用的特性，即可以配置Watch窗口，以便始终都能显示线程的最后错误代码的号码和该错误的英文描述。通过选定Watch窗口中的一行，并键入“@err,hr”，就能够做到这一点。观察图1-1，你会看到已经调用了CreateFile函数。该函数返回INVALID_HANDLE_VALUE（-1）的HANDLE，表示它未能打开指定的文件。但是Watch窗口向我们显示最后错误代码（即如果调用GetLastError函数，该函数返回的错误代码）是0x00000002。该Watch窗口又进一步指明错误代码2是指“系统不能找到指定的文件。”你会发现它与WinError.h头文件中的错误代码2所指的字符串是相同的。

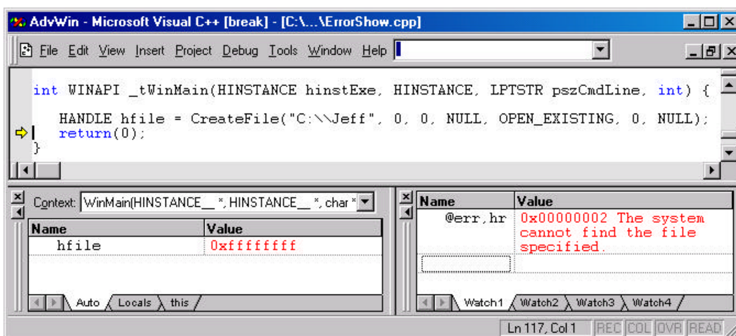

图1-1 在Visual Studio 6.0的Watch窗口中键入“@err,hr”，就可以查看当前线程的最后错误代码

China-pub.com 下载

Visual studio还配有一个小的实用程序，称为Error Lookup。可以使用Error Lookup将错误代码的号码转换成相应文本描述（见图1-2）。

如果在编写的应用程序中发现一个错误，可能想要向用户显示该错误的文本描述。Windows提供了一个函数，可以将错误代码转换成它的文本描述。该函数称为 Format-Message，显示如下：

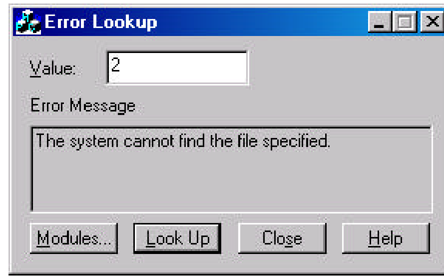

图1-2 Error Lookup窗口

```
DWORD FormatMessage(
   DWORD dwFlags,
   LPCVOID pSource,
   DWORD dwMessageId,
   DWORD dwLanguageId,
   PTSTR pszBuffer,
   DWORD nSize,
   va_list *Arguments);
```

FormatMessage函数的功能实际上是非常丰富的，在创建向用户显示的字符串信息时，它是首选函数。该函数之所以有这样大的作用，原因之一是它很容易用多种语言进行操作。该函数能够检测出用户首选的语言（在Regional Settings Control Panel小应用程序中设定），并返回相应的文本。当然，首先必须自己转换字符串，然后将已转换的消息表资源嵌入你的 .exe文件或DLL模块中，然后该函数会选定正确的嵌入对象。 ErrorShow示例应用程序（本章后面将加以介绍）展示了如何调用该函数，以便将Microsoft公司定义的错误代码转换成它的文本描述。

有些人常常问我，Microsoft公司是否建立了一个主控列表，以显示每个 Windows函数可能返回的所有错误代码。可惜，回答是没有这样的列表，而且 Microsoft公司将永远不会建立这样的一个列表。因为在创建系统的新版本时，建立和维护该列表实在太困难了。

建立这样一个列表存在的问题是，你可以调用一个 Windows函数，但是该函数能够在内部调用另一个函数，而这另一个函数又可以调用另一个函数，如此类推。由于各种不同的原因，这些函数中的任何一个函数都可能运行失败。有时，当一个函数运行失败时，较高级的函数对它进行恢复，并且仍然可以执行你想执行的操作。为了创建该主控列表， Microsoft公司必须跟踪每个函数的运行路径，并建立所有可能的错误代码的列表。这项工作很困难。而且，当创建系统的新版本时，这些函数的运行路径还会改变。

1.1 定义自己的错误代码

前面已经说明 Windows函数是如何向函数的调用者指明发生的错误，你也能够将该机制用于自己的函数。比如说，你编写了一个希望其他人调用的函数，你的函数可能因为这样或那样的原因而运行失败，你必须向函数的调用者说明它已经运行失败。

若要指明函数运行失败，只需要设定线程的最后的错误代码，然后让你的函数返回FALSE、INVALID_HANDLE_VALUE、NULL或者返回任何合适的信息。若要设定线程的最后错误代码，只需调用下面的代码：

请将你认为合适的任何 32位号码传递给该函数。尝试使用 WinError.h中已经存在的代码，

```
VOID SetLastError(DWORD dwErrCode);
```

China-pub.com
下载

只要该代码能够正确地指明想要报告的错误即可。如果你认为 WinError.h中的任何代码都不能正确地反映该错误的性质，那么可以创建你自己的代码。错误代码是个 32位的数字，划分成表1-2 所示的各个域。

表1-2 错误代码的域

位	31~30	29	28	27~16	15~0
内容	严重性	Microsoft/客户	保留	设备代码	异常代码
含义	0=成功 1=供参考 2=警告 3=错误	0=Microsoft公司定义的代码 1=客户定义的代码	必须是0	由Microsoft公司定义	由Microsoft/客户定义

这些域将在第24章中详细讲述。现在，需要知道的重要域是第 29位。Microsoft公司规定，他们建立的所有错误代码的这个信息位均使用 0。如果创建自己的错误代码，必须使 29位为1。这样，就可以确保你的错误代码与Microsoft公司目前或者将来定义的错误代码不会发生冲突。

1.2 ErrorShow示例应用程序

ErrorShow应用程序“ 01 ErrorShow.exe ”(在清单1-1中列出)展示了如何获取错误代码的文本描述的方法。该应用程序的源代码和资源文件位于本书所附光盘上的 01-ErrorShow目录下。一般来说，该应用程序用于显示调试程序的 Watch窗口和Error Lookup程序是如何运行的。当启动该程序时，就会出现如图1-3所示的窗口。

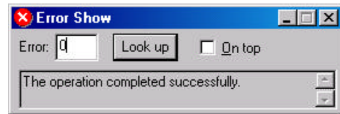

图1-3 Error Show窗口

可以将任何错误代码键入该编辑控件。当单击 Look up按钮时，在底部的滚动窗口中就会显示该错误的文本描述。该应用程序唯一令人感兴趣的特性是如何调用 FormatMessage函数。下面是使用该函数的方法：

```
// Get the error code
DWORD dwError = GetDlgItemInt(hwnd, IDC_ERRORCODE, NULL, FALSE);

HLOCAL hlocal = NULL;   // Buffer that gets the error message string

// Get the error code's textual description
BOOL fOk = FormatMessage(
   FORMAT_MESSAGE_FROM_SYSTEM | FORMAT_MESSAGE_ALLOCATE_BUFFER,
   NULL, dwError, MAKELANGID(LANG_ENGLISH, SUBLANG_ENGLISH_US),
   (LPTSTR) &hlocal, 0, NULL);
   :
if (hlocal != NULL) {
   SetDlgItemText(hwnd, IDC_ERRORTEXT, (PCTSTR) LocalLock(hlocal));
   LocalFree(hlocal);
} else {
   SetDlgItemText(hwnd, IDC_ERRORTEXT, TEXT("Error number not found."));
}
```

第一个代码行用于从编辑控件中检索错误代码的号码。然后，建立一个内存块的句柄并将它初始化为NULL。FormatMessage函数在内部对内存块进行分配，并将它的句柄返回给我们。

China-pub.com 下载

当调用FormatMessage函数时，传递了FORMAT_MESSAGE_FROM_SYSTEM标志。该标志告诉FormatMessage函数，我们想要系统定义的错误代码的字符串。还传递了 FORMAT_MESSAGE_ALLOCATE_BUFFER标志，告诉该函数为错误代码的文本描述分配足够大的内存块。该内存块的句柄将在hlocal变量中返回。第三个参数指明想要查找的错误代码的号码，第四个参数指明想要文本描述使用什么语言。

如果FormatMessage函数运行成功，那么错误代码的文本描述就位于内存块中，将它拷贝到对话框底部的滚动窗口中。如果FormatMessage函数运行失败，设法查看NetMsg.dll模块中的消息代码，以了解该错误是否与网络有关。使用 NetMsg.dll模块的句柄，再次调用FormatMessage函数。你会看到，每个DLL（或.exe）都有它自己的一组错误代码，可以使用Message Compiler（MC.exe）将这组错误代码添加给该模块，并将一个资源添加给该模块。这就是Visual Studio的Error Lookup工具允许你用Modules对话框进行的操作。以下是清单1-1 ErrorShow示例应用程序。

清单1-1 ErrorShow示例应用程序

ErrorShow.cpp

```
/******************************************************************************
Module:  ErrorShow.cpp
Notices: Copyright (c) 2000 Jeffrey Richter
******************************************************************************/


#include "..\CmnHdr.h"     /* See Appendix A. */
#include <Windowsx.h>
#include <tchar.h>
#include "Resource.h"


///////////////////////////////////////////////////////////////////////////////


#define ESM_POKECODEANDLOOKUP    (WM_USER + 100)
const TCHAR g_szAppName[] = TEXT("Error Show");


///////////////////////////////////////////////////////////////////////////////


BOOL Dlg_OnInitDialog(HWND hwnd, HWND hwndFocus, LPARAM lParam) {

   chSETDLGICONS(hwnd, IDI_ERRORSHOW);

   // Don't accept error codes more than 5 digits long
   Edit_LimitText(GetDlgItem(hwnd, IDC_ERRORCODE), 5);

   // Look up the command-line passed error number
   SendMessage(hwnd, ESM_POKECODEANDLOOKUP, lParam, 0);
   return(TRUE);
}
```

China-pub.com
下载


```
///////////////////////////////////////////////////////////////////////////////


void Dlg_OnCommand(HWND hwnd, int id, HWND hwndCtl, UINT codeNotify) {
switch (id) {

case IDCANCEL:
   EndDialog(hwnd, id);
   break;

case IDC_ALWAYSONTOP:
   SetWindowPos(hwnd, IsDlgButtonChecked(hwnd, IDC_ALWAYSONTOP)
      ? HWND_TOPMOST : HWND_NOTOPMOST, 0, 0, 0, 0, SWP_NOMOVE | SWP_NOSIZE);
   break;

case IDC_ERRORCODE:
   EnableWindow(GetDlgItem(hwnd, IDOK), Edit_GetTextLength(hwndCtl) > 0);
   break;

case IDOK:
   // Get the error code
   DWORD dwError = GetDlgItemInt(hwnd, IDC_ERRORCODE, NULL, FALSE);

   HLOCAL hlocal = NULL;   // Buffer that gets the error message string

   // Get the error code's textual description
   BOOL fOk = FormatMessage(
      FORMAT_MESSAGE_FROM_SYSTEM | FORMAT_MESSAGE_ALLOCATE_BUFFER,
      NULL, dwError, MAKELANGID(LANG_ENGLISH, SUBLANG_ENGLISH_US),
      (PTSTR) &hlocal, 0, NULL);

   if (!fOk) {
      // Is it a network-related error?
      HMODULE hDll = LoadLibraryEx(TEXT("netmsg.dll"), NULL,
         DONT_RESOLVE_DLL_REFERENCES);

      if (hDll != NULL) {
         FormatMessage(
            FORMAT_MESSAGE_FROM_HMODULE | FORMAT_MESSAGE_FROM_SYSTEM,
            hDll, dwError, MAKELANGID(LANG_ENGLISH, SUBLANG_ENGLISH_US),
            (PTSTR) &hlocal, 0, NULL);
         FreeLibrary(hDll);
      }
   }

   if (hlocal != NULL) {
      SetDlgItemText(hwnd, IDC_ERRORTEXT, (PCTSTR) LocalLock(hlocal));
      LocalFree(hlocal);
         } else {
            SetDlgItemText(hwnd, IDC_ERRORTEXT, TEXT("Error number not found."));
         }
         break;
      }
   }

///////////////////////////////////////////////////////////////////////////////
```

China-pub.com 下载

```
INT_PTR WINAPI Dlg_Proc(HWND hwnd, UINT uMsg, WPARAM wParam, LPARAM lParam) {

   switch (uMsg) {
      chHANDLE_DLGMSG(hwnd, WM_INITDIALOG, Dlg_OnInitDialog);
      chHANDLE_DLGMSG(hwnd, WM_COMMAND,    Dlg_OnCommand);

   case ESM_POKECODEANDLOOKUP:
      SetDlgItemInt(hwnd, IDC_ERRORCODE, (UINT) wParam, FALSE);
      FORWARD_WM_COMMAND(hwnd, IDOK, GetDlgItem(hwnd, IDOK), BN_CLICKED,
         PostMessage);
      SetForegroundWindow(hwnd);
      break;
   }

   return(FALSE);
}


///////////////////////////////////////////////////////////////////////////////


int WINAPI _tWinMain(HINSTANCE hinstExe, HINSTANCE, PTSTR pszCmdLine, int) {

   HWND hwnd = FindWindow(TEXT("#32770"), TEXT("Error Show"));
   if (IsWindow(hwnd)) {
      // An instance is already running, activate it and send it the new #
      SendMessage(hwnd, ESM_POKECODEANDLOOKUP, _ttoi(pszCmdLine), 0);
   } else {
      DialogBoxParam(hinstExe, MAKEINTRESOURCE(IDD_ERRORSHOW),
         NULL, Dlg_Proc, _ttoi(pszCmdLine));
   }
   return(0);
}

//////////////////////////////// End of File //////////////////////////////////
//Microsoft Developer Studio generated resource script.
//
#include "resource.h"

#define APSTUDIO_READONLY_SYMBOLS
/////////////////////////////////////////////////////////////////////////////
//
// Generated from the TEXTINCLUDE 2 resource.
//
#include "afxres.h"

/////////////////////////////////////////////////////////////////////////////
#undef APSTUDIO_READONLY_SYMBOLS

/////////////////////////////////////////////////////////////////////////////
// English (U.S.) resources

#if !defined(AFX_RESOURCE_DLL) || defined(AFX_TARG_ENU)
#ifdef _WIN32
LANGUAGE LANG_ENGLISH, SUBLANG_ENGLISH_US
#pragma code_page(1252)
#endif //_WIN32
```

China-pub.com
下载


```
////////////////////////////////////////////////////////////////////////////
//
// Dialog
//

IDD_ERRORSHOW DIALOGEX 0, 0, 182, 42
STYLE DS_SETFOREGROUND | DS_3DLOOK | DS_CENTER | WS_MINIMIZEBOX | WS_VISIBLE |
    WS_CAPTION | WS_SYSMENU
CAPTION "Error Show"
FONT 8, "MS Sans Serif"
BEGIN
    LTEXT           "Error:",IDC_STATIC,4,4,19,8
    EDITTEXT        IDC_ERRORCODE,24,2,24,14,ES_AUTOHSCROLL | ES_NUMBER
    DEFPUSHBUTTON   "Look up",IDOK,56,2,36,14
    CONTROL         "&On top",IDC_ALWAYSONTOP,"Button",BS_AUTOCHECKBOX |
                    WS_TABSTOP,104,4,38,10
    EDITTEXT        IDC_ERRORTEXT,4,20,176,20,ES_MULTILINE | ES_AUTOVSCROLL |
                    ES_READONLY | NOT WS_BORDER | WS_VSCROLL,
                    WS_EX_CLIENTEDGE
END


////////////////////////////////////////////////////////////////////////////
//
// DESIGNINFO
//

#ifdef APSTUDIO_INVOKED
GUIDELINES DESIGNINFO DISCARDABLE
BEGIN
    IDD_ERRORSHOW, DIALOG
    BEGIN
        LEFTMARGIN, 7
        RIGHTMARGIN, 175
        TOPMARGIN, 7
        BOTTOMMARGIN, 35
    END
END
#endif    // APSTUDIO_INVOKED


#ifdef APSTUDIO_INVOKED
////////////////////////////////////////////////////////////////////////////
//
// TEXTINCLUDE
//

1 TEXTINCLUDE DISCARDABLE
BEGIN
    "resource.h\0"
END

2 TEXTINCLUDE DISCARDABLE
BEGIN
```

China-pub.com 下载

```
        "#include ""afxres.h""\r\n"
        "\0"
    END

    3 TEXTINCLUDE DISCARDABLE
  BEGIN
      "\r\n"
      "\0"
  END

  #endif    // APSTUDIO_INVOKED


  /////////////////////////////////////////////////////////////////////////////
  //
  // Icon
  //

  // Icon with lowest ID value placed first to ensure application icon
  // remains consistent on all systems.
  IDI_ERRORSHOW            ICON    DISCARDABLE     "ErrorShow.ico"
  #endif    // English (U.S.) resources
  /////////////////////////////////////////////////////////////////////////////



  #ifndef APSTUDIO_INVOKED
  /////////////////////////////////////////////////////////////////////////////
  //
  // Generated from the TEXTINCLUDE 3 resource.
  //


  /////////////////////////////////////////////////////////////////////////////
  #endif    // not APSTUDIO_INVOKED
```