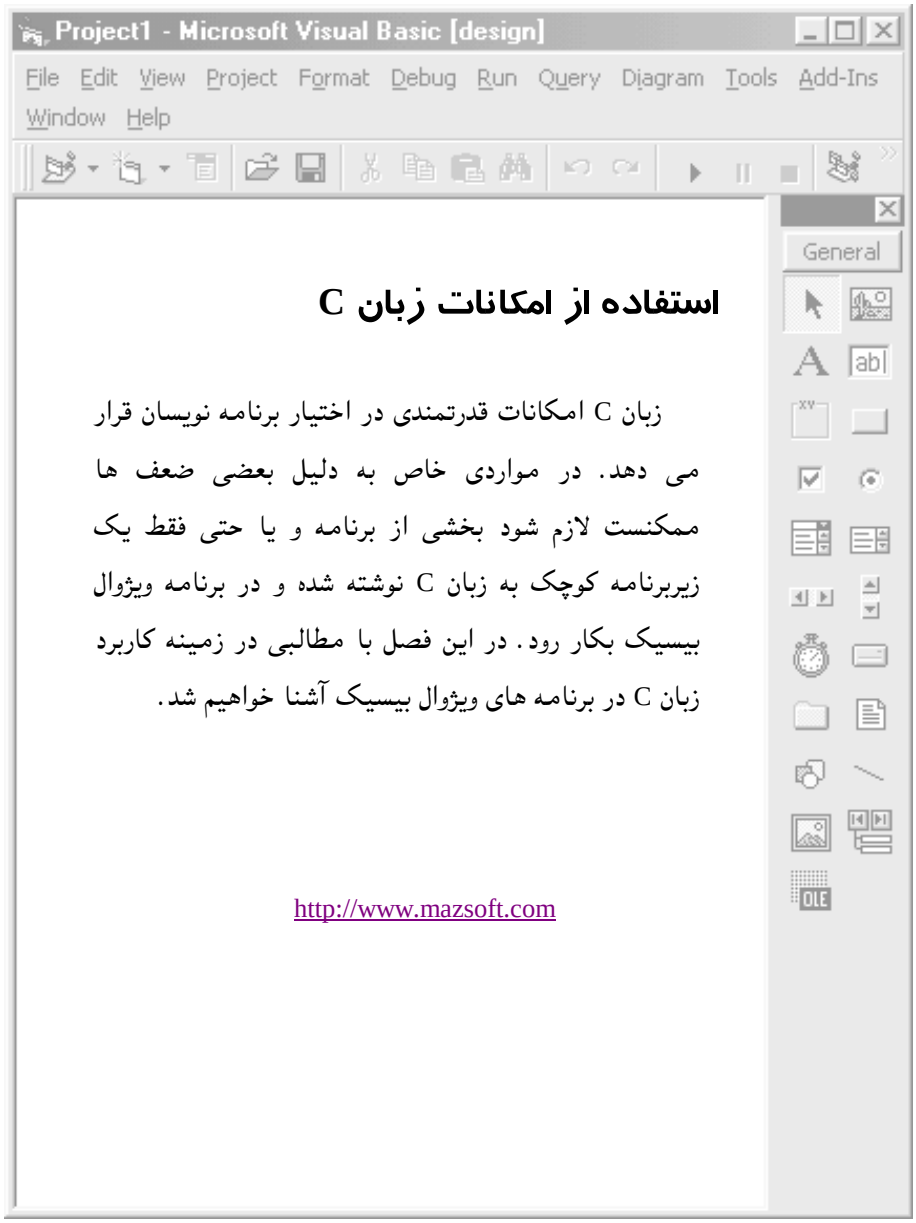

Project1 - Microsoft Visual Basic [design]
File Edit View Project Format Debug Run Query Diagram Tools Add-Ins
Window Help
General
استفاده از امکانات زبان C
زبان C امکانات قدرتمندی در اختیار برنامه نویسان قرار می دهد. در مواردی خاص به دلیل بعضی ضعف ها ممکنست لازم شود بخشی از برنامه و یا حتی فقط یک زیربرنامه کوچک به زبان C نوشته شده و در برنامه ویژوال بیسیک بکار رود. در این فصل با مطالبی در زمینه کاربرد زبان C در برنامه های ویژوال بیسیک آشنا خواهیم شد.
http://www.mazsoft.com

شبیه سازی ماکروهای HIWORD ، LOWORD ، HIBYTE و LOBYTE

نام فایل	تاریخ	نگارش ویژوال بیسیک	نگارش ویندوز
112651.TXT	۲۱ آگوست ۱۹۹۵	All	All

ویژوال بیسیک توابعی برای عملیات روی بیت ها در اختیار نمی گذارد. در زبان C ماکروهایی (HIWORD، LOWORD، HIBYTE و LOBYTE) وجود دارند که برای تقسیم اعداد صحیح بزرگ[۱] یا اعداد صحیح معمولی به دو قسمت، به کار می روند. در این بخش با ارایه یک مثال نحوه انجام این کار در برنامه های ویژوال بیسیک نشان داده می شود.

ماکروهای HIWORD، LOWORD، HIBYTE و LOBYTE در زبان C در فایل WINDOWS.H تعریف شده اند. ماکروهای HIBYTE و LOBYTE برای استخراج بایت بالا یا بایت پایین عدد صحیح معمولی ارسالی برای آنها بکار می روند. ماکروهای HIWORD و LOWORD دوبایت بالا یا دوبایت پایین را از عدد صحیح بزرگی که برایشان ارسال می شود استخراج می کنند.

۱- یک پروژه جدید در ویژوال بیسیک ایجاد کنید. Form1 بطور پیش فرض ایجاد میرود.

۲- دو CommandButton روی فرم قرار دهید.

۳- کد زیر را به واقعه Command1_Click اضافه کنید:

```
Sub Command1_Click ()
   Dim wParam As Integer
   Dim LOBYTE As Integer
   Dim HIBYTE As Integer
   ' Set wParam to a value:
   wParam = &H77FF
   ' Make call to function:
   ret = gethilobyte(wParam, LOBYTE, HIBYTE)
   ' Print out return values:
   Print LOBYTE, HIBYTE
End Sub
```

۴- کد زیر را به واقعه Command2_Click اضافه کنید:

---

[۱] Long Integer

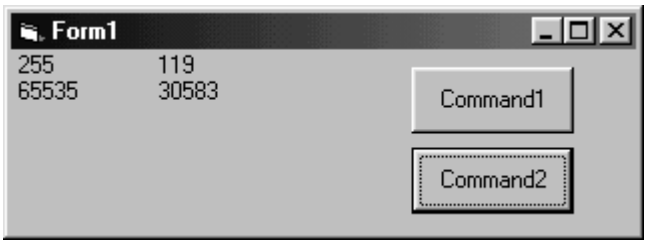
```
Sub Command2_Click ()
   Dim lParam As Long
   Dim LOWORD As Long
   Dim HIWORD As Long
   ' Set lParam to a value:
   lParam = &H7777FFFF
   ' Make call to function:
   ret = gethiloword(lParam, LOWORD, HIWORD)
   ' Print out return values:
   Print LOWORD, HIWORD
End Sub
```

۵- کد زیر را در بخش Declarations از Form1 وارد کنید:

```
' Enter the following Function statement as one, single line:
Function gethilobyte(wparam as integer, LOBYTE as integer,
   HIBYTE as integer)
   ' This is the LOBYTE of the wParam:
   LOBYTE = wParam And  &HFF&
   ' This is the HIBYTE of the wParam:
   HIBYTE = wParam  \  &H100  And  &HFF&
   gethilobyte = 1
End Function

Function gethiloword(lparam as long, LOWORD as long, HIWORD as long)
   ' This is the LOWORD of the lParam:
   LOWORD = lParam And  &HFFFF&
   ' LOWORD now equals 65,535 or &HFFFF
   ' This is the HIWORD of the lParam:
   HIWORD = lParam  \  &H10000  And  &HFFFF&
   ' HIWORD now equals 30,583 or &H7777
   gethiloword = 1
End Function
```

۶- برنامه را اجرا کنید. روی Command1 کلیک کنید تا یک مقدار عدد صحیح به بایت بالا و بایت پایین تقسیم شود. روی Command2 کلیک کنید تا یک مقدار عدد صحیح بزرگ به دو بایت بالا و و دو بایت پایین تقسیم شود.



شکل 1 - نتیجه اجرای مثال

ردوبدل کردن پارامترهای اختیاری بین ویژوال بیسیک و توابع C

نام فایل	تاریخ	نگارش ویژوال بیسیک	نگارش ویندوز
153546.TXT	۲۴ آگوست ۱۹۹۸	4 , 5 , 6	95 / 98 / NT

این امکان وجود دارد که بتوان برای یک تابع C موجود در یک DLL پارامترهایی را بصورت اختیاری تعریف کرد. ویژوال بیسیک دارای یک کلمه کلیدی به نام Optional است که می توان از آن در دستور Declare استفاده نمود. این عبارت به کامپایلر اطلاع می دهد که اگر پارامتر مربوطه از فراخوانی تابع حذف شد، یک واریانت از نوع VT_ERROR برای تابع ارسال کند. در غیر اینصورت اگر پارامتر حذف نشد، همانطور که تعریف شده است آنرا ارسال کند. در هر صورت، در سمت تابع C هیچ چیزی اختیاری نیست - تابع همیشه تعداد پارامترهای معینی که برایش تعریف شده است را دریافت می کند.

مثال زیر نشان می دهد چگونه برای یک تابع C پارامترهای اختیاری ارسال کنید.

۱- یک DLL ۳۲ بیتی مخصوص ویندوز، حاوی تابع زیر در C ایجاد کنید:

```
long _stdcall OptionalParamCall(LPSTR pStr, VARIANT op1, VARIANT op2)
{
   if (op1.vt == VT_ERROR && op1.scode == DISP_E_PARAMNOTFOUND)
   MessageBox (NULL, "Optional Param1 is Empty!", "Test DLL", MB_OK);

   if (op2.vt == VT_ERROR && op2.scode == DISP_E_PARAMNOTFOUND)
   MessageBox (NULL, "Optional Param2 is Empty!", "Test DLL", MB_OK);

   MessageBox (NULL, pStr, "Test DLL", MB_OK);
   return 1;
}
```

۲- تابع را در یک فایل DEF. بصورت زیر تعریف کنید:

```
LIBRARY   TESTDLL

CODE      PRELOAD MOVEABLE DISCARDABLE
DATA      PRELOAD SINGLE

EXPORTS
OptionalParamCall     @1
```

این DLL را Testdll.dll نام گذاری کرده و فایل را در شاخه system\ (برای ویندوز ۹۵ و ۹۸) یا system32\ (برای ویندوز NT) قرار دهید.

۳- در ویژوال بیسیک یک پروژه جدید ایجاد کنید و کد زیر را در بخش Declarations از Form1 وارد کنید:

```
Private Declare Function OptionalParamCall Lib "testdll.dll" _
   (ByVal s As String, Optional ByVal op1, Optional ByVal op2) As Long

   Private Sub Form_Click()
      ret& = OptionalParamCall("hello")
      ret& = OptionalParamCall("hello", 7)
      ret& = OptionalParamCall("hello", , "world")
      ret& = OptionalParamCall("hello", 8.2, "Mike")
   End Sub
```

۴- برنامه ویژوال بیسیک را اجرا کنید، و روی فرم کلیک کنید. یک سری پنجره های پیام ظاهر می شوند که نشان دهنده این هستند که کدام پارامترها خالی ارسال شده اند و مقدار رشته ارسالی به عنوان اولین پارامتر را نشان می دهد.

فراخوانی توابع ویژوال بیسیک از درون یک DLL

نام فایل	تاریخ	نگارش ویژوال بیسیک	نگارش ویندوز
181578.TXT	۲۴ آگوست ۱۹۹۸	5 , 6	95 / 98 / NT

با استفاده از اپراتور جدید AddressOf در ویژوال بیسیک می توانید اشاره گر به یک تابع ویژوال بیسیک را برای یک DLL ارسال کنید و DLL را مجبور کنید تابع را از طریق آن اشاره گر فرابخواند. در این بخش مثالی ارایه می شود که علاوه بر نحوه کاربرد این تکنیک، نحوه ارسال رشته ها از DLL به ویژوال بیسیک را نیز نمایش می دهد.

برای ساخت مثال زیر مراحل قدم به قدم گفته شده در آنرا انجام دهید تا یک پروژه EXE ویژوال بیسیک و یک پروژه DLL ویژوال C++ ایجاد شوند. سپس برنامه ویژوال بیسیک خود را اجرا کنید تا مثال مورد آزمایش قرار گیرد.

مراحل ساخت بخش ویژوال بیسیک مثال

۱- یک پروژه EXE جدید در ویژوال بیسیک ایجاد کنید.

۲- یک Command Button به Form1 اضافه کنید.

۳- دستور Declare زیر را به بخش Declarations از Form1 اضافه کنید:

```
Private Declare Sub ExecuteCallback Lib "vcvbdll" ( _
   ByVal pFunc as Long)
```

۴- کد زیر را به واقعه کلیک از Command Button اضافه کنید:

```
Call ExecuteCallback(AddressOf MyCallback)
```

۵- یک ماژول جدید به پروژه اضافه کنید و کد زیر را در آن وارد کنید:

```
Sub MyCallback(ByVal parm as String)
   MsgBox "You are inside the VB callback function!"
   MsgBox "Parameter passed in was: " & parm
End Sub
```

مراحل ساخت بخش ویژوال C++ مثال

۱- در ویژوال C++ به کمک MFC AppWizard یک پروژه DLL جدید به نام cvcbdll ایجاد کرده و تمامی گزینه ها را به همان صورت پیش فرض تعریف کنید.

۲- کد زیر را به انتهای فایل vcvbdll.cpp اضافه کنید:

```
void __stdcall ExecuteCallback(long cbAddress) {
   // Declare the function pointer, with one string argument.
   typedef void (__stdcall *FUNCPTR)(BSTR pbstr);
   FUNCPTR vbFunc;

   // Point the function pointer at the passed-in address.
   vbFunc = (FUNCPTR)cbAddress;

   // Call the function through the function pointer.
   vbFunc(SysAllocString(L"Hi! This message came from your DLL!"));
}
```

۳- سطر زیر را به انتهای فایل vcvbdll.def اضافه کنید تا تابع ExecuteCallback برای دیگر برنامه ها قابل استفاده باشد:

```
ExecuteCallback
```

پروژه را کامپایل کنید و فایل DLL. را در شاخه Windows\System\ کپی و آنرا آزمایش کنید.

مراجع

برای کسب اطلاعات بیشتر می توانید مراجع زیر را مطالعه کنید:

```
Microsoft Visual Basic 5.0 Books Online: AddressOf

ARTICLE-ID: Q171729
TITLE     : HOWTO: Do Generic Callbacks Using a Helper DLL
```

رد و بدل کردن رشته بین ویژوال بیسیک و DLL زبان C

نام فایل	تاریخ	نگارش ویژوال بیسیک	نگارش ویندوز
187912.TXT	۱۷ آگوست ۱۹۹۸	5 , 6	95 / 98 / NT

هنگام ایجاد یک کتابخانه DLL به زبان C برای استفاده با ویژوال بیسیک، باید به یاد داشت که C و ویژوال بیسیک با رشته ها به نحو متفاوتی رفتار می کنند. برای اطمینان از اینکه همه رشته ها بطور صحیح ارسال می شوند باید تمامی کارهای احتیاطی لازم را انجام دهید، در غیر اینصورت ممکنست یک اشکال اساسی در کار ویندوز پیش بیاید. در این بخش خواهید دید که چگونه یک تابع به زبان C در یک DLL بوجود آورید که رشته ای را بصورت پارامتر دریافت می کند و نیز روش فراخوانی آن تابع در ویژوال بیسیک را نیز طی مثالی ملاحظه خواهید نمود.

برای ایجاد DLL به زبان C باید بدانید که چگونه رشته ها برای توابع DLL شما ارسال می شوند و به همین دلیل دانستن اینکه ویژوال بیسیک چگونه بطور داخلی با رشته ها کار می کند بسیار مهم است. بیشتر توابع C انتظار دارند رشته ها بصورت آرایه ای از کاراکترهای ASCII باشند که با null خاتمه یافته اند. اما ویژوال بیسیک از یک نوع خاص رشته به نام

BSTR استفاده می کند. در محیط Win32 یک BSTR اشاره گری است به بخشی از حافظه که حاوی یک رشته کاراکتری از نوع Unicode و یک پیشوند از نوع عدد صحیح ۳۲ بیتی برای تعیین طول رشته می باشد.

بیشتر این تفاوت ها قابل چشم پوشی است، چرا که ویژوال بیسیک تمامی BSTR ها را پیش از ارسال به توابع غیر ویژوال بیسیک به ASCII تبدیل می کند. مشکل اصلی مربوط به خود پارامتر است. ویژوال بیسیک، به طور پیش فرض، تمامی متغیرها از جمله رشته ها را بصورت ارجاعی ارسال می کند. از آنجا که یک متغیر BSTR خود یک اشاره گر به یک رشته است، هنگامی که ویژوال بیسیک آنرا بصورت ارجاعی ارسال می کند در عمل اشاره گری به یک اشاره گر دیگر به رشته مورد نظر را ارسال می کند. بیشتر توابع C انتظار دارند یک اشاره گر به یک رشته برایشان ارسال شود (مانند LPSTR).

مراحل قدم به قدم برای ساخت بخش C مثال

۱- ویژوال C++ را باز کرده و از منوی File گزینه New را انتخاب کنید. در تابلوی Projects گزینه Win32 Dynamic-Link Library را انتخاب کرده و نام پروژه را StrSamp قرار دهید.

۲- یکبار دیگر از منوی File گزینه New را انتخاب کرده و در تابلوی Files گزینه C++ Source File را انتخاب کنید و نام فایل را "StrSamp.c" تعیین کنید و OK را فشار دهید.

۳- مرحله ۲ را تکرار کنید و این بار Text File را به عنوان نوع فایل انتخاب کنید. نام فایل را "StrSamp.def" قرار داده و OK را فشار دهید.

۴- کد زیر را به StrSamp.c اضافه کنید:

```
#include <windows.h>

void __stdcall DisplayStringByVal(LPCSTR pszString)
{
   //pszString is a pointer to a string
   MessageBox(NULL, pszString, "Display String ByVal",
              MB_OK | MB_ICONINFORMATION);
}

void __stdcall DisplayStringByRef(LPCSTR* ppszString)
```

```
{
   //ppszSting is a pointer to a pointer to a string
   MessageBox(NULL, *ppszString, "Display String ByRef",
             MB_OK | MB_ICONINFORMATION);
}

void __stdcall FillString(LPSTR pszString)
{
   // Create a temp buffer with our string
   char buffer[] = "Hello from the C DLL!";

   // Copy our temp string to pszString
   strcpy(pszString, buffer);
}

int __stdcall InStrRev(LPCSTR pszString, short iChar)
{
   // This function is similar to Visual Basic's InStr function
   // except that it searches for the given ASCII character from
   // right to left, returning the character position of the
   // last occurrence (rather than the first) of the character
   // in the string.
   char* pszTmp;
   int nRet = 0;

   // Scan for iChar in pszString backwards
   pszTmp = strrchr(pszString, (int)iChar);
   if(pszTmp != NULL)
      nRet = pszTmp - pszString + 1;

   return nRet;
}
```

۵- برای اینکه توابع را بتوان از بیرون از DLL فراخواند سطرهای زیر را به فایل StrSamp.def اضافه کنید:

```
LIBRARY StrSamp
DESCRIPTION 'Microsoft KB Sample DLL'
EXPORTS
   DisplayStringByVal
   DisplayStringByRef
   FillString
   InStrRev
```

۶- توسط منوی Build پروژه DLL خود را کامپایل کنید. هنگامی که فایل DLL ساخته شد، آنرا برای آزمایش در دایرکتوری ویژوال بیسیک خود کپی کنید.

مراحل ایجاد بخش ویژوال بیسیک مثال

۱- یک پروژه معمولی در ویژوال بیسیک ایجاد کنید و نامش را StrTest بگذارید. Form1 بطور پیش فرض ایجاد می شود.

۲- سه Command Button روی Form1 قرار دهید.

۳- کد زیر را در بخش Declarations از Form1 وارد کنید:

```
Option Explicit

Private Declare Sub DisplayStringByRef Lib "StrSamp.dll" _
   (sMyString As String)
Private Declare Sub DisplayStringByVal Lib "StrSamp.dll" _
   (ByVal sMyString As String)
Private Declare Sub FillString Lib "StrSamp.dll" _
   (ByVal sMyString As String)
Private Declare Function InStrRev Lib "StrSamp.dll" _
   (ByVal sMyString As String, ByVal iChar As Integer) _
   As Long
```

توجه کنید که بیشتر رشته ها بصورت ByVal (مقداری) تعریف شده اند. این بدین معنا نیست که عملاً مقدار رشته ها ارسال می شوند، بلکه متغیر BSTR بصورت مقداری ارسال می شود (به یاد دارید که یک متغیر BSTR یک اشاره گر است به یک رشته). به این ترتیب عبارت ByVal باعث می شود یک اشاره گر دور که به یک رشته اشاره می کند (LPSTR) ارسال شود، که دقیقاً همان چیزی است که تابع C انتظار دارد.

۴- کد زیر را به واقعه کلیک از هر یک از Command Button ها اضافه کنید:

```
Private Sub Command1_Click()
   Dim sTestString1 As String
   Dim sTestString2 As String

   sTestString1 = "This is my string passed to the dll by value."
   DisplayStringByVal sTestString1

   sTestString2 = "This is my string passed to the dll by reference."
   DisplayStringByRef sTestString2

End Sub

Private Sub Command2_Click()
   Dim sFillTest As String

   sFillTest = Space$(260)
   FillString sFillTest
   MsgBox Trim$(sFillTest), vbInformation, "Fill String"
```

```
End Sub

Private Sub Command3_Click()
     Dim sPathString As String
     Dim sMsg As String
     Dim lCharPosition As Long

     sPathString = "C:\My Documents\Temp\Item.txt"
     lCharPosition = InStrRev(sPathString, Asc("\"))

     If CBool(lCharPosition) Then
        sMsg = "The file '" & Mid$(sPathString, lCharPosition + 1)
        sMsg = sMsg & "' is at this location:" & vbCrLf & vbCrLf
        sMsg = sMsg & Left$(sPathString, lCharPosition - 1)
        MsgBox sMsg, vbInformation, "InStrRev"
     Else
        MsgBox "Cannot find '/' in " & sPathString, vbCritical
    End If

End Sub
```

۵- برای اجرای برنامه در محیط ویژوال بیسیک، کلید F5 را فشار دهید.

☜ توجه: اگر پیغام خطایی دریافت کردید، ممکنست به خاطر این باشد که ویژوال بیسیک نمی تواند DLL را پیدا کند. پیش از اجرای مثال، مطمئن شوید که DLL مورد نیاز را در شاخه ویژوال بیسیک کپی کرده اید.

ارسال آرایه از نوع داده تعریف شده توسط کاربر، برای C

نام فایل	تاریخ	نگارش ویژوال بیسیک	نگارش ویندوز
194609.TXT	۲۰ نوامبر ۱۹۹۸	5 , 6	95 , 98 , NT

هنگامی که با آرایه ای از نوع داده تعریف شده توسط کاربر[۲] (UDT) در ویژوال بیسیک کار می کنید ممکن است بخواهید آن آرایه را به یک DLL نوشته شده توسط C/C++ ارسال کنید. چنین آرایه ای را نمی توان مستقیماً ارسال نمود، زیرا آرایه هایی از نوع SAFEARRAY به

[۲] User Defined Type (UDT)

داده هایی از نوع Automation-Safe محدود می شوند و UDT از انواع استاندارد Automation نمی باشد.

علاوه بر این، اگر UDT حاوی رشته هایی با طول متغیر باشد، نمی توانید با ارسال یک اشاره گر به اولین عضو از آرایه مشکل را حل نمایید، زیرا ویژوال بیسیک به منظور انجام تبدیلات رشته های Unicode به ANSI یک کپی موقتی از عضو مذکور ایجاد می کند. در این بخش روشی معرفی می گردد که به کمک آن بر این مشکلات فایق آمده و آرایه ای از نوع UDT که حاوی رشته با طول متغیر باشد را برای DLL های نوشته شده توسط C/C++ ارسال نمایید.

راه حل ارایه شده در اینجا، برای جلوگیری از اینکه ویژوال بیسیک یک کپی از محتویات آرایه ایجاد کند و رشته ها را از Unicode به ANSI تبدیل کند، از کتابخانه نوع[۳] کمک می گیرد. برای کسب اطلاعات بیشتر در این مورد به مراجع ذکر شده در انتهای این بخش مراجعه کنید.

نوع داده تعریف شده توسط کاربر (UDT) در این مثال به این شکل است:

```
Type TestUDT
   l As Long
   str As String
End Type
```

مراحل ساخت DLL و کتابخانه نوع مربوطه

۱- ویژوال C++ را باز کنید و از منوی File گزینه New را انتخاب کنید. در تابلوی Project گزینه Win32 Dynamic-Link Library را انتخاب کرده و نام پروژه را UDTArray قرار دهید.

۲- مجدداً از منوی File گزینه New را انتخاب کنید. در تابلوی Files گزینه C++ Source File را انتخاب کنید و نام فایل را UDTArray.c وارد کرده و OK را فشار دهید.

[۳] Type Library

۳- مرحله ۲ را تکرار کرده و این بار نوع فایل را Text File انتخاب کنید. نام فایلها را به ترتیب UDTArray.def و UDTArray.odl بگذارید.

۴- سپس کد زیر را در UDTArray.c وارد کنید:

```
#include <windows.h>
#include <oleauto.h>

struct tagTestUDT
{
   long l;
   BSTR str;
} TestUDT;

void __stdcall ModifyStruct(struct tagTestUDT *t, long nTotalItem)
{
   // modify the last element
   long i = nTotalItem - 1;  // the array begins from 0

   (t+i)->l = 200;

   SysFreeString((t+i)->str );  // free the string passed in
   (t+i)->str = SysAllocString(OLESTR("Changed."));
}
```

۵- برای این که بتوان توابع را از بیرون از DLL فراخواند، سطرهای زیر را در فایل UDTArray.def وارد کنید:

```
EXPORTS
  ModifyStruct
```

۶- با اضافه کردن سطرهای زیر در UDTArray.odl توابع خود را در یک کتابخانه نوع تعریف کنید:

```
[
  uuid(D6229080-2CC3-11d2-9FD8-00C04F8EF4D4),
  helpstring("Pass Array of UDTs Helper")
]

library PassUDTLib
{
  //definition of the UDT
  typedef struct tagTestUDT
  {
    long l;
    BSTR str;
  }TestUDT;

  [
     helpstring("test"),
     dllname("UDTArray.dll")
  ]
```

```
    module structDLL
    {
       [
         helpstring("Modify the elements of array"),
         entry("ModifyStruct")
       ]
       void _stdcall ModifyStruct
       (
         [in, out] TestUDT* t,
         [in] long nTotalItem
       );
    };
  };
```

۷- از منوی Project گزینه Settings را انتخاب کنید و تابلوی C/C++ را کلیک کنید. سپس از لیست کشویی Category ، مورد Code Generation را انتخاب کنید. Struct Member Alignment را به 4 Bytes تغییر دهید زیرا ویژوال بیسیک از صف های ۴ بایتی استفاده می کند در صورتیکه در ویژوال C++ مقدار پیش فرض ۸ بایت است.

۸- بوسیله منوی Build و گزینه Rebuild All فایل DLL خود را ایجاد کنید. سپس فایل حاصل را (UDTArray.dll) به شاخه ویژوال بیسیک کپی کنید تا مورد آزمایش قرار گیرد.

مراحل ایجاد بخش ویژوال بیسیک مثال

۹- ویژوال بیسیک را اجرا کرده و یک پروژه معمولی EXE ایجاد کنید. Form1 بطور پیش فرض بوجود می آید.

۱۰- از منوی Project گزینه References را انتخاب کنید تا پنجره References ظاهر شود، سپس روی Browse کلیک کنید. فایل UDTArray.tlb را که برای DLL شما ایجاد شده بود پیدا کنید.

۱۱- یک Command Button به Form1 اضافه کنید و کد زیر را در Form1 وارد نمایید:

```
Private Sub Command1_Click()
    ' Note: You do not need to redefine TestUDT because it is in
    ' the type library we added to our project.
    Dim t(0 To 1) As TestUDT
    Dim i as long

    t(0).l = 1
    t(0).str = "test1"
```

```
    t(1).l = 2
    t(1).str = "Test2"

    i = UBound(t) - LBound(t) + 1
    ModifyStruct t(0), i  't(0)'s address will be passed to C

    MsgBox "t(1).l = " & t(1).l & vbCrLf & "t(1).str = " & t(1).str
End Sub
```

۱۲- برنامه ویژوال بیسیک را اجرا کنید و روی Command Button کلیک کنید. پنجره پیامی ظاهر می شود که نشان دهنده اینست که آخرین عضو از آرایه تغییر یافته است.

مراجع

برای کسب اطلاعات کاملی در زمینه ساخت DLL به زبان C و استفاده از آن در ویژوال بیسیک به مراجع زیر مراجعه کنید:

- DLLs for Beginners در کتابخانه شبکه برنامه سازان مایکروسافت (MSDN).
- فایل VB5DLL.DOC موجود در دایرکتوری Tools\Docs از سی دی رام ویژوال بیسیک ۵.
- برای کسب اطلاعات بیشتر در مورد کاربرد کتابخانه نوع در تعریف توابع C/C++ در ویژوال بیسیک به بخش زیر از بانک اطلاعاتی مایکروسافت رجوع کنید:

```
ARTICLE-ID: Q189133
TITLE   : HOWTO: Make C DLL More Accessible to VB with a Type Library
```

- برای اطلاعات بیشتر به بخشهای زیر از بانک اطلاعاتی مایکروسافت رجوع کنید:

```
ARTICLE-ID: Q142840
TITLE     : Visual Basic Requirements for Exported DLL Functions

ARTICLE-ID: Q171583
TITLE     : HOWTO: Fill a 32-bit VBA Array of UDType via a
            Visual C++ DLL
```