<http://www.csharphelp.com/archives/archive52.html>

**Call Unmanaged Code. Part 1 - Simple DLLImport**  
By [Vyacheslav Biktagirov](http://www.csharphelp.com/bio/vbiktagirov.html)

|  |
| --- |
| Managed world is beautiful, I have all classes I want in FrameWork.. But what if I want call some unmanaged code? For instance, I have DLL written in C++, and want use it from C#.  Let's look some code. Our DLL exports some function, in CDecl convention, that sums two integers:  extern "C" \_\_declspec(dllexport) \_\_cdecl int sum(int a,int b);  And, of course, we want reuse this code in C#. We must recall, that it is no "direct" way to call unmanaged code, but we must inform the compiler, what we want to call, how, and where is needed code located.  [DllImport("TestDll.dll", EntryPoint="sum",  ExactSpelling=false,CallingConvention=CallingConvention.Cdecl)]  static extern int sum(int a,int b);  and now we can call it like normal C# function.  x=5;  y=7;  z=sum(x,y); // x will receive 12  Here is full C# client code - tested for Beta2. |

using System;

using System.Drawing;

using System.Collections;

using System.ComponentModel;

using System.[Windows[http://images.intellitxt.com/ast/adTypes/mag-glass_10x10.gif](http://www.csharphelp.com/archives/archive52.html)](http://www.csharphelp.com/archives/archive52.html).Forms;

using System.Data;

using System.Runtime.InteropServices;

namespace WindowsApplication6

{

///

/// Summary description for Form1.

///

public class Form1 : System.Windows.Forms.Form

{

private System.Windows.Forms.Button button1;

private System.Windows.Forms.TextBox textBox1;

private System.Windows.Forms.Label label1;

private System.Windows.Forms.TextBox textBox2;

private System.Windows.Forms.Label label2;

private System.Windows.Forms.TextBox textBox3;

///

/// Required designer variable.

///

private System.ComponentModel.Container components = null;

public Form1()

{

//

// Required for Windows Form Designer support

//

InitializeComponent();

//

// TODO: Add any constructor code after InitializeComponent call

//

}

///

/// Clean up any resources being used.

///

protected override void Dispose( bool disposing )

{

if( disposing )

{

if (components != null)

{

components.Dispose();

}

}

base.Dispose( disposing );

}

#region Windows Form Designer generated code

///

/// Required method for Designer support - do not modify

/// the contents of this method with the code editor.

///

private void InitializeComponent()

{

this.button1 = new System.Windows.Forms.Button();

this.textBox1 = new System.Windows.Forms.TextBox();

this.label1 = new System.Windows.Forms.Label();

this.textBox2 = new System.Windows.Forms.TextBox();

this.label2 = new System.Windows.Forms.Label();

this.textBox3 = new System.Windows.Forms.TextBox();

this.SuspendLayout();

//

// button1

//

this.button1.Location = new System.Drawing.Point(64, 192);

this.button1.Name = "button1";

this.button1.Size = new System.Drawing.Size(144, 64);

this.button1.TabIndex = 0;

this.button1.Text = "call sum";

this.button1.Click += new System.EventHandler(this.button1\_Click);

//

// textBox1

//

this.textBox1.Location = new System.Drawing.Point(40, 120);

this.textBox1.Name = "textBox1";

this.textBox1.Size = new System.Drawing.Size(72, 22);

this.textBox1.TabIndex = 1;

this.textBox1.Text = "2";

//

// label1

//

this.label1.Location = new System.Drawing.Point(128, 128);

this.label1.Name = "label1";

this.label1.Size = new System.Drawing.Size(16, 16);

this.label1.TabIndex = 2;

this.label1.Text = "+";

//

// textBox2

//

this.textBox2.Location = new System.Drawing.Point(152, 120);

this.textBox2.Name = "textBox2";

this.textBox2.Size = new System.Drawing.Size(56, 22);

this.textBox2.TabIndex = 3;

this.textBox2.Text = "3";

//

// label2

//

this.label2.Location = new System.Drawing.Point(224, 120);

this.label2.Name = "label2";

this.label2.Size = new System.Drawing.Size(24, 23);

this.label2.TabIndex = 4;

this.label2.Text = "=";

//

// textBox3

//

this.textBox3.Location = new System.Drawing.Point(248, 120);

this.textBox3.Name = "textBox3";

this.textBox3.Size = new System.Drawing.Size(112, 22);

this.textBox3.TabIndex = 5;

this.textBox3.Text = "5";

//

// Form1

//

this.AutoScaleBaseSize = new System.Drawing.Size(6, 15);

this.ClientSize = new System.Drawing.Size(576, 322);

this.Controls.AddRange(new System.Windows.Forms.Control[] {this.textBox3,this.label2,this.textBox2,this.label1,this.textBox1,this.button1});

this.Name = "Form1";

this.Text = "Form1";

this.ResumeLayout(false);

}

#endregion

///

/// The main entry point for the application.

///

[STAThread]

static void Main()

{

Application.Run(new Form1());

}

#region My Code

#region Dll Imports

[DllImport("TestDll.dll", EntryPoint="sum",

ExactSpelling=false,CallingConvention=CallingConvention.Cdecl)]

static extern int sum(int a,int b);

#endregion

#region Button Click Events

private void button1\_Click(object sender, System.EventArgs e)

{

textBox3.Text=(int.Parse(textBox1.Text)+int.Parse(textBox2.Text)).ToString();

}

#endregion

#endregion

}

}

It sounds very simple, becouse "int" is isomorphic type, says, int in C# and ind C++ is identical. What we can do, when we want operate non-isomorhic types, like String? Recall, that .NET string is some Class, while C++ string is char\*,or wchar\_t\*,or BSTR, .. String may be embedded in a structure, or pointed by pointer, or even something more exotic. Let's call some string function.

[DllImport("Advapi32.dll", EntryPoint="GetUserName", ExactSpelling=false,

SetLastError=true)]

static extern bool GetUserName(

[MarshalAs(UnmanagedType.LPArray)] byte[] lpBuffer,

[MarshalAs(UnmanagedType.LPArray)] Int32[] nSize );

This function receives two parameters: char\* and int\*. Becouse we must allocate char\* buffer and receive string by pointer, we can't use UnmanagedType.LPStr attribute, so we pass ANSI string as byte array. int\* is more simple-it's 1-element Int32 array. Let's call it:

private void button2\_Click(object sender, System.EventArgs e)

{

byte[] str=new byte[20];

Int32[] len=new Int32[1];

len[0]=20;

GetUserName(str,len);

MessageBox.Show(System.Text.Encoding.ASCII.GetString(str));

}

We allocate 20 bytes for receiving ANSI string,one element in Int32 array, set 20 as max string length and call it. For receiving string from byte array I used Text.Encoding.ASCII class.

That's enough for first part. Second part will speak about more complex interop.