OSInfo Sample

Article02/01/2013

This sample demonstrates how to pass either a formatted class by value or a structure by reference as a parameter to an unmanaged function that expects a structure with an embedded character buffer.

The OSInfo sample uses the following unmanaged function, shown with its original function declaration:

- **GetVersionEx** exported from Kernel32.dll.

```
// BOOL GetVersionEx(LPOSVERSIONINFO lpVersionInfo);
```

The original structure passed to the function contains the following elements:

```
typedef struct _OSVERSIONINFO
{
  DWORD dwOSVersionInfoSize;
  DWORD dwMajorVersion;
  DWORD dwMinorVersion;
  DWORD dwBuildNumber;
  DWORD dwPlatformId;
  TCHAR szCSDVersion[ 128 ];
} OSVERSIONINFO;
```

In this sample, the OSVersionInfo class and the OSVersionInfo2 structure produce the same results when passed to the unmanaged function. The MarshalAsAttribute attribute sets the UnmanagedType enumeration value to **ByValTStr**, which is used to identify the inline, fixed-length character array that appears within the unmanaged structure.

The LibWrap class contains two prototypes: GetVersionEx passes the class as a parameter and GetVersionEx2 passes a structure as a parameter. By applying the InAttribute and OutAttribute attributes explicitly, the sample ensures that OSVersionInfo is marshaled as an In/Out parameter and the caller can see the changes marshaled back. (For performance, the default directional attribute for a class is In, which prevents the caller from seeing the changes marshaled back.)

The OSVersionInfo2 structure, which is normally passed by value, is declared with the **ref** (**ByRef** in Visual Basic) keyword and is passed by reference. The Marshal.SizeOf method determines the size, in bytes, of the unmanaged structure.

Declaring Prototypes

C#

```
[StructLayout(LayoutKind.Sequential)]
public class OSVersionInfo
{
    public int OSVersionInfoSize;
    public int MajorVersion;
    public int MinorVersion;
    public int BuildNumber;
    public int PlatformId;

    [MarshalAs(UnmanagedType.ByValTStr, SizeConst=128)]
    public String CSDVersion;
}

[StructLayout(LayoutKind.Sequential)]
public struct OSVersionInfo2
{
    public int OSVersionInfoSize;
    public int MajorVersion;
    public int MinorVersion;
    public int BuildNumber;
    public int PlatformId;

    [MarshalAs(UnmanagedType.ByValTStr, SizeConst=128)]
    public String CSDVersion;
}

public class LibWrap
{
    [DllImport("kernel32")]
    public static extern bool GetVersionEx([In, Out] OSVersionInfo osvi);

    [DllImport("kernel32", EntryPoint="GetVersionEx")]
    public static extern bool GetVersionEx2(ref OSVersionInfo2 osvi);
}
```

Calling Functions

C#

```
public class App
{
    public static void Main()
    {
        Console.WriteLine("\nPassing OSVersionInfo as a class");

        OSVersionInfo osvi = new OSVersionInfo();
        osvi.OSVersionInfoSize = Marshal.SizeOf(osvi);

        LibWrap.GetVersionEx(osvi);

        Console.WriteLine("Class size:    {0}", osvi.OSVersionInfoSize);
        Console.WriteLine("OS Version:    {0}.{1}", osvi.MajorVersion,
osvi.MinorVersion);

        Console.WriteLine("\nPassing OSVersionInfo as a struct" );

        OSVersionInfo2 osvi2 = new OSVersionInfo2();
        osvi2.OSVersionInfoSize = Marshal.SizeOf(osvi2);

        LibWrap.GetVersionEx2(ref osvi2);
        Console.WriteLine("Struct size:   {0}", osvi2.OSVersionInfoSize);
        Console.WriteLine("OS Version:    {0}.{1}", osvi2.MajorVersion,
osvi2.MinorVersion);
    }
}
```

See Also

Concepts

Marshaling Strings

Platform Invoke Data Types

Default Marshaling for Strings

Other Resources

Creating Prototypes in Managed Code