

Addressing reference to struct in C#

Source:

<http://www.tech-archive.net/Archive/DotNet/microsoft.public.dotnet.framework.interop/2004-11/0404.html>

From: Lau Lei Cheong (*leu_lc_at_yahoo.com.hk*)

Date: 11/24/04

Date: Wed, 24 Nov 2004 18:18:19 +0800

Hello,

It happens that one days I had to write program that uses Windows API, but seems C# do not have anything as convenient as "windows.h" in C++, so I had to declare the functions myself.

While most of the codes I found on the net involving C++, I have to do some ports in order to use them in my code. (and actually I'm building wrapper classes for the APIs, based on functionality) And I have some problem on the declarations.

I'll use ReadFile() to illustrate the problem, here's the prototype defined in MSDN:

```
BOOL ReadFile(HANDLE hFile, LPVOID lpBuffer, DWORD nNumberOfBytesToRead, LPDWORD lpNumberOfBytesRead, LPOVERLAPPED lpOverlapped);
```

And on the web, I see variants of declarations:

```
[DllImport("kernel32.dll", SetLastError=true)]
static extern int ReadFile (int hFile, ref object lpBuffer, int
nNumberOfBytesToRead, ref int lpNumberOfBytesRead, ref OVERLAPPED
lpOverlapped);
```

```
[DllImport("kernel32.dll", SetLastError=true)]
static extern int ReadFile (int hFile, ref object lpBuffer, int
nNumberOfBytesToRead, out int lpNumberOfBytesRead, out OVERLAPPED
lpOverlapped);
```

```
[DllImport("kernel32.dll", SetLastError=true)]
public static extern bool ReadFile(System.IntPtr hFile, byte[] lpBuffer,
uint nNumberOfBytesToRead, out uint lpNumberOfBytesRead, out uint
lpOverlapped);
```

```
[DllImport("kernel32.dll", SetLastError=true)]
public static extern bool ReadFile(System.IntPtr hFile, byte[] lpBuffer,
uint nNumberOfBytesToRead, out int lpNumberOfBytesRead, System.IntPtr
```

lpOverlapped);

Okay, it seems that the first and last one is closer to definition, so I included both.

It becomes strange that when places that C++ code uses null and C# yields compile time error when I use that. I have to use IntPtr.Zero in this case. Is there any equivalent to the "null in C++ " in C#. In this way I won't have to duplicate function declarations just to work around cases I have to pass nulls to them.

I don't what to use unsafe keyword because of some issues. (One of them is I have poor mind over pointers. :P)

I feel that there should be neater way addressing this issue. Please help if anyone knows. Thank you.

Regards,
Lau Lei Cheong

--

If you want to mail to me, please replace all instance of "e" to "a" in my email address. Thank you.

Sorry for any inconvenience caused, but there's many spam-mailers.