

Re: MprAdminInterfaceEnum

Source:

<http://www.tech-archive.net/Archive/DotNet/microsoft.public.dotnet.framework.interop/2004-12/0017.html>

From: Juan Ignacio Gelos (*msnews_at_juanignaciogelos.com*)

Date: 12/01/04

Date: Wed, 1 Dec 2004 04:54:08 -0300

Found the answer here:

<http://www.dotnetinterop.com/faq/?q=CalleeAllocatedStructArray>

Regards,
Juan

"Juan Ignacio Gelos" <msnews@juanignaciogelos.com> wrote in message
news:Ob5P9pu1EHA.1452@TK2MSFTNGP11.phx.gbl...

> *Hi,*
>
> *Trying to enumerate interfaces on RRAS.*
> *Bellow is what I have so far. The MprAdminInterfaceEnum has an*
> *lpLpbBuffer out parameter which is a buffer that will contain an array of*
> *MPR_INTERFACE_0's.*
> *I want to access the wszInterfaceName for each interface. How should I*
> *marshal this parameter?*
>
> *Thanks,*
> *Juan*
>
> */// <summary>*
> */// The MprAdminInterfaceEnum function enumerates all the interfaces on*
> *a*
> *specified server.*
> */// </summary>*
> */// <param name="hMprServer">[in] Handle to the router on which to*
> *execute*
> *this call. Obtain this handle by calling MprAdminServerConnect.</param>*
> */// <param name="dwLevel">[in] Specifies the level of information passed*
> *in lpBuffer. Must be zero.</param>*
> */// <param name="lpLpbBuffer">[out] Pointer to a pointer variable that*
> *points to an array of MPR_INTERFACE_0 structures on successful return.*
> *This*
> *memory must be freed by a call to the MprAdminBufferFree function.</param>*
> */// <param name="dwPrefMaxLen">[in] Specifies the preferred maximum*
> *length*
> *of returned data (in 8-bit bytes). If this parameter is -1, the buffer*

> returned is large enough to hold all available information. </param>
> /// <param name="lpdwEntriesRead">[out] Pointer to a DWORD variable.
This
> variable receives the total number of interfaces that were enumerated from
> the current position in the enumeration.</param>
> /// <param name="lpdwTotalEntries">[out] Pointer to a DWORD variable.
This
> variable receives the total number of interfaces that could have been
> enumerated from the current resume position.</param>
> /// <param name="lpdwResumeHandle">[in] Pointer to a DWORD variable.
This
> variable specifies a resume handle that can be used to continue the
> enumeration. The handle should be zero on the first call, and left
unchanged
> on subsequent calls. If the return code is ERROR_MORE_DATA then the call
can
> be re-issued using the handle to retrieve more data. If on return, the
> handle is NULL, the enumeration cannot be continued. For other types of
> error returns, this handle is invalid.
> /// This parameter is optional. If the calling application specifies
NULL
> for this parameter, the function does not return a resume handle.</param>
> /// <returns></returns>
> /// <remarks><code>
> /// DWORD APIENTRY
> /// MprAdminInterfaceEnum(
> /// IN MPR_SERVER_HANDLE hMprServer,
> /// IN DWORD dwLevel,
> /// OUT LPBYTE * lpLpbBuffer,
> /// IN DWORD dwPrefMaxLen,
> /// OUT LPDWORD lpdwEntriesRead,
> /// OUT LPDWORD lpdwTotalEntries,
> /// IN LPDWORD lpdwResumeHandle OPTIONAL
> ///);
> /// </code></remarks>
> [DllImport("Mprapi.dll", CharSet = CharSet.Ansi)]
> public static extern int MprAdminInterfaceEnum(
> IntPtr hMprServer,
> [MarshalAs(UnmanagedType.U4)]
> int dwLevel,
>
> ref MPR_INTERFACE_0[] lpLpbBuffer,
> int dwPrefMaxLen,
> ref int lpdwEntriesRead,
> ref int lpdwTotalEntries,
> ref int lpdwResumeHandle
>);
>
>
>
> [StructLayout(LayoutKind.Sequential, CharSet = CharSet.Ansi)]

```
> public struct MPR_INTERFACE_0
> {
> [MarshalAs(UnmanagedType.ByValTStr, SizeConst =
> MprApi.MAX_INTERFACE_NAME_LEN+1)]
> public string wszInterfaceName;
> public IntPtr hInterface;
> public bool fEnabled;
> public ROUTER_INTERFACE_TYPE dwIfType;
> public ROUTER_CONNECTION_STATE dwConnectionState;
> public UnreachabilityReasons fUnReachabilityReasons;
> public int dwLastError;
> }
>
>
```