

waveOutPrepareHeader crashes with Vista x64

Source:

<http://www.tech-archive.net/Archive/DotNet/microsoft.public.dotnet.framework.interop/2007-06/msg00070.html>

- *From:* "assaf" <assafwo@xxxxxxxxxx>
 - *Date:* Wed, 13 Jun 2007 13:55:20 +0300
-

Hello all.

I am invoking waveOutPrepareHeader in Vista x64.

I am getting the following error:

"An invalid parameter was passed to a system function."

Here is the code:

```
using System;
using System.Collections.Generic;
using System.Text;
using System.Runtime.InteropServices;
using System.Diagnostics;

namespace ConsoleApplication1
{
    class Program
    {
        [StructLayout(LayoutKind.Sequential)]
        internal struct WAVEFORMATEX
        {
            internal ushort wFormatTag; /* format type */
            internal ushort nChannels; /* number of channels (i.e. mono, stereo...) */
            internal uint nSamplesPerSec; /* sample rate */
            internal uint nAvgBytesPerSec; /* for buffer estimation */
            internal ushort nBlockAlign; /* block size of data */
            internal ushort wBitsPerSample; /* number of bits per sample of mono data */
            internal ushort cbSize; /* the count in bytes of the size of */
            /* extra information (after cbSize) */
        }

        internal enum Callbacks
        {
            EVENT = 0x00050000,
            NULL = 0x00000000, /* no callback */
            WINDOW = 0x00010000, /* dwCallback is a HWND */
            TASK = 0x00020000, /* dwCallback is a HTASK */
            FUNCTION = 0x00030000 /* dwCallback is a FARPROC */
        }
    }
}
```

waveOutPrepareHeader crashes with Vista x64

```
}

[DllImport("winmm")]
internal static extern
uint waveOutOpen(
    out int phwi,
    uint uDeviceID,
    ref WAVEFORMATEX pwf,
    uint threadId,
    int dwCallbackInstance,
    Callbacks fdwOpen
);

[DllImport("kernel32")]
internal static extern
int GetCurrentThreadId();

[DllImport("Winmm")]
internal static extern
uint waveOutGetErrorTextA(
    uint mmrError,
    byte[] pszText,
    int cchText
);

private static void Assert(uint res)
{
    if (res != 0)
    {
        byte[] buff = new byte[1000];
        res = waveOutGetErrorTextA(res, buff, buff.Length);
        Assert(res);
    }
}

int c = Array.IndexOf<byte>(buff, 0);
string s = ASCIIEncoding.ASCII.GetString(buff, 0, c);
throw new ApplicationException(s);
}
}

[DllImport("winmm")]
internal static extern
uint waveOutPrepareHeader(
    int hwi,
    ref WAVEHDR pwh,
    int cbwh
);

[StructLayout(LayoutKind.Sequential)]
internal struct WAVEHDR
{
    internal uint lpData; /* 32-bit pointer to locked data buffer */
}
```

waveOutPrepareHeader crashes with Vista x64

```
internal uint dwBufferLength; /* length of data buffer */
internal uint dwBytesRecorded; /* used for input only */
internal int dwUser; /* for client's use */
internal uint dwFlags; /* assorted flags (see defines) */
internal uint dwLoops; /* loop control counter */
int lpNext; /* reserved for driver */
int reserved; /* reserved for driver */
}

[DllImport("winmm")]
internal static extern
uint waveOutWrite(
int hwi,
ref WAVEHDR pwh,
int cbwh
);

[DllImport("winmm")]
internal static extern
uint waveOutClose(
int hwi
);

static void Main(string[] args)
{
int hWaveIn;

WAVEFORMATEX wf;
wf.cbSize = 0;
wf.nAvgBytesPerSec = 8000;
wf.nBlockAlign = 1;
wf.nChannels = 1;
wf.nSamplesPerSec = 8000;
wf.wBitsPerSample = 8;
wf.wFormatTag = 1;

uint Id = (uint)GetCurrentThreadId();

uint res = waveOutOpen(out hWaveIn, 0, ref wf, Id, 0, Callbacks.TASK);
Assert(res);

uint size = 1000;
WAVEHDR wh = new WAVEHDR();
wh.dwUser = 0;
wh.dwBufferLength = size;
wh.dwBytesRecorded = 0;
wh.dwFlags=0;
wh.dwLoops=0;
IntPtr i = Marshal.AllocHGlobal((int)size);
Debug.Assert( (long)i <= uint.MaxValue, "(long)i > uint.MaxValue");
wh.lpData = (uint)i;
}
```

waveOutPrepareHeader crashes with Vista x64

```
int whSize = Marshal.SizeOf(wh);

res = waveOutPrepareHeader(hWaveIn, ref wh, whSize);
Assert(res); // An invalid parameter was passed to a system function.

res = waveOutWrite(hWaveIn, ref wh, whSize);
Assert(res);

res = waveOutClose(hWaveIn);
Assert(res);
}
}
}
```