

Re: QASetWindowsJournalHook by managed code

Source:

<http://www.tech-archive.net/Archive/DotNet/microsoft.public.dotnet.framework.compactframework/2009-04/msg000>

- *From:* "Chris Tacke, eMVP" <ctacke.at.opennetcf.dot.com>
 - *Date:* Sun, 5 Apr 2009 14:15:15 -0500
-

Again, WTF are you using all of these GCHandles for? The fact you keep throwing them (and more and more it seems) indicates that you clearly don't understand them. To improve your development skills, you need to **understand** the code you write. You need to understand that an IntPtr and int and a GCHandle are all actually very closely related items (and in fact so is a delegate). They are all simply 32-bit integers. Managed code gives them different significance, and how the GC treats them varies slightly, but it's really all syntactic sugar around a very, very basic and fundamental thing.

Let me be clear – I'm not going to just write and test this code for you. I don't have the time or the inclination – I already work long days and the only reason I do stuff that isn't related to something I need to be doing is if I find it interesting. I did hooking a long time ago and got it working, so it no longer really interests me.

The confusion you're having is one of the main reasons I think that to be a good managed developer you still **must** understand C. You need to understand, and understand well, how memory works. This is an excellent opportunity for you to actually learn what is happening and why what you have is failing. A good step would be to write what you want to happen in native code and get that working. Then translate what you have in native code to managed code. I still do this exercise a lot when writing managed calls to native stuff, especially if it's complex, so I can use the Memory View and see if the addresses getting passed around make sense in both managed and native. If you're not doing that, then you need to learn how or this kind of problem will always be a black box to you and your development skills will never progress beyond mediocre.

—

Chris Tacke, Embedded MVP
OpenNETCF Consulting
Giving back to the embedded community
<http://community.OpenNETCF.com>

Re: QASetWindowsJournalHook by managed code

"Jack P" <private@xxxxxxxxxxxx> wrote in message
news:uXnRaWhJHA.5376@xxxxxxxxxxxxxxxxxxxxxxxxxxxx

I feel so frustrated, it just can't work.
and
I compared your code, I only found the differences below:

1. QASetWindowsJournalHook return IntPtr, my is return Int.
2. JournalHookStruct , my is a class.
3. delegate int HookProc, my is public

I don't think they are problems, however, I still tried everything became the same as yours, but still can't work (will crash after executed any other program)

Here is my latest code, could you help to review ? appreciate.

```
////////////////////////////////////  
using System;  
using System.Linq;  
using System.Collections.Generic;  
using System.ComponentModel;  
using System.Data;  
using System.Drawing;  
using System.Text;  
using System.Windows.Forms;  
using System.Runtime.InteropServices;  
  
namespace JB  
{  
  
    public partial class Form1 : Form  
    {  
        public struct JournalHookStruct  
        {  
            public int message;  
            public int paramL;  
            public int paramH;  
            public int time;  
            public IntPtr hwnd;  
        }  
  
        public static IntPtr hHook;  
        public static IntPtr GwParam; // for MouseHookProc  
        public static IntPtr GIParam; // for MouseHookProc  
        public delegate int HookProc(int nCode, IntPtr wParam, IntPtr lParam);  
        public static JournalHookStruct myHookMessage = new JournalHookStruct();  
        public static HookProc MouseHookProcedure = new
```

Re: QASetWindowsJournalHook by managed code

```
HookProc(MouseHookProc);
public static IntPtr pMouseHookProcedure =
Marshal.GetFunctionPointerForDelegate(MouseHookProcedure);

/// <summary>
/// try to keep everything alive
/// </summary>
public static GCHandle gc1 = GCHandle.Alloc(myHookMessage,
GCHandleType.Pinned);
public static GCHandle gc2 = GCHandle.Alloc(MouseHookProcedure,
GCHandleType.Pinned);
public static GCHandle gc3 = GCHandle.Alloc(pMouseHookProcedure,
GCHandleType.Pinned);
public static GCHandle gc4 = GCHandle.Alloc(hHook,
GCHandleType.Pinned);
public static GCHandle gc5 = GCHandle.Alloc(GwParam,
GCHandleType.Pinned);
public static GCHandle gc6 = GCHandle.Alloc( lParam,
GCHandleType.Pinned);

public static int MouseHookProc(int nCode, IntPtr wParam, IntPtr
lParam)
{
while (false)
{
int a = 1;
}
GwParam = wParam;
lParam = lParam;
return NativeDLLs.CallNextHookEx(gc4.AddrOfPinnedObject(),
nCode, gc5.AddrOfPinnedObject(), gc6.AddrOfPinnedObject()); //
pass wParam / lParam directly, the result is the same.
}

public Form1()
{
InitializeComponent();
hHook = NativeDLLs.QASetWindowsJournalHook(0,
pMouseHookProcedure, gc1.AddrOfPinnedObject()); // whatever pass
MouseHookProcedure , address of gc2, target of gc3, result are the same
=> will crash after executed other program
}
}

public class NativeDLLs
{
public enum HookType
{
JournalRecord = 0,
JournalPlayback = 1,
KeyboardLowLevel = 20
}
```

Re: QASetWindowsJournalHook by managed code

```
}

[DllImport("coredll.dll", SetLastError = true)]
public static extern IntPtr GetModuleHandle(string mod);

[DllImport("coredll.dll", SetLastError = true)]
public static extern IntPtr QASetWindowsJournalHook(HookType
nFilterType, IntPtr pfnFilterProc, IntPtr pfnEventMsg);

[return: MarshalAs(UnmanagedType.Bool)]
[DllImport("CoreDll.dll", EntryPoint =
"QAUnhookWindowsJournalHook", SetLastError = true)]
public static extern int QAUnhookWindowsJournalHook(IntPtr hook);

[DllImport("CoreDll.dll", CharSet =
CharSet.Auto/*,CallingConvention = CallingConvention.StdCall*/)
public static extern int CallNextHookEx(IntPtr idHook, int nCode,
IntPtr wParam, IntPtr lParam);
}
}
```

"Chris Tacke, eMVP" <ctacke.at.opennetcf.dot.com> wrote in message
news:etAMs1XtJHA.3816@xxxxxxxxxxxxxxxxxxxxxxxxxxxx

Just digging though code I have, I see this:

```
#if !DESKTOP
[DllImport("coredll.dll", SetLastError = true)]
public static extern IntPtr GetModuleHandle(string mod);

[DllImport("coredll.dll", SetLastError = true)]
public static extern IntPtr QASetWindowsJournalHook(HookType
nFilterType, HookProc pfnFilterProc, ref JournalHookStruct pfnEventMsg);

/*
HHOOK
WINAPI
QASetWindowsJournalHook(
int nFilterType,
HOOKPROC pfnFilterProc,
EVENTMSG *pfnEventMsg
);
*/
#endif

/*
```

Re: QASetWindowsJournalHook by managed code

```
* From pwinuser.h
*
typedef struct tagEVENTMSG {
UINT message;
UINT paramL;
UINT paramH;
DWORD time;
HWND hwnd;
} EVENTMSG, *PEVENTMSGMSG, NEAR *NPEVENTMSGMSG, FAR
*LPEVENTMSGMSG;
*/
public struct JournalHookStruct
{
public int message;
public int paramL;
public int paramH;
public int time;
public IntPtr hwnd;
}

/*
* From pwinuser.h
*
#define WH_JOURNALRECORD 0
#define WH_JOURNALPLAYBACK 1
#define WH_KEYBOARD_LL 20
*/
internal enum HookType
{
JournalRecord = 0,
JournalPlayback = 1,
KeyboardLowLevel = 20
}

internal delegate int HookProc(HookCode nCode, IntPtr wParam, IntPtr
lParam);
```

That should get you a long ways there.

Chris Tacke, Embedded MVP
OpenNETCF Consulting
Giving back to the embedded community
<http://community.OpenNETCF.com>

"Jack P" <private@xxxxxxxxxxxx> wrote in message
<news:OZ86VzVtJHA.1088@xxxxxxxxxxxxxxxxxxxxxxxxxxxx>

Re: QASetWindowsJournalHook by managed code

could you teach me where is wrong ?

I almost try everything out

"Chris Tacke, eMVP" <ctacke.at.opennetcf.dot.com> wrote in message
news:un7lxxStJHA.4364@xxxxxxxxxxxxxxxxxxxxxxxxxxxxx

Now it looks like you're just throwing code at the wall hoping something will stick. You need to go back and reasearch what a GCHandle is and understand when you should and shouldn't use them. You seem to just have them in here to have them. You also need to see how GetFunctionPointerForDelegate works and exactly what it returns. You need to **understand** the code, not just grasp at straws.

--

Chris Tacke, Embedded MVP
OpenNETCF Consulting
Giving back to the embedded community
<http://community.OpenNETCF.com>

"Jack P" <private@xxxxxxxxxxxx> wrote in message
news:O%23xELRLtJHA.1236@xxxxxxxxxxxxxxxxxxxxxxxxxxxxx

Hi Chris,
I added few lines as following for GetFunctionPointerForDelegate:

```
public static IntPtr  
pMouseHookProcedure =  
Marshal.GetFunctionPointerForDelegate(MouseHookProcedure);  
public static GCHandle gc3  
=  
GCHandle.Alloc(pMouseHookProcedure,  
GCHandleType.Pinned);
```

Re: QASetWindowsJournalHook by managed code

and changed the init_hook()

as following:

```
public static void
```

```
init_hook() ////////////////
```

```
change
```

```
(HookProc) to (IntPtr)
```

```
because can't convert IntPtr
```

```
back to HookProc)
```

```
/////
```

```
{
```

```
hHook =
```

```
LibWrap.QASetWindowsJournalHook(0,
```

```
pMouseHookProcedure,
```

```
gc1.AddrOfPinnedObject());
```

```
}
```

It didn't work yet (behave

the same as before, system

will crash

after execute other program

)

and here is my complete

code:

```
////////////////////////////////////
```

```
using System;
```

```
using
```

```
System.Collections.Generic;
```

```
using System.Linq;
```

```
using System.Text;
```

```
using
```

```
System.Runtime.InteropServices;
```

```
using
```

```
System.Security.Permissions;
```

```
namespace ClassLibrary1
```

```
{
```

```
unsafe public delegate int
```

```
HookProc(int nCode, IntPtr
```

```
wParam,
```

```
IntPtr lParam);
```

```
[SecurityPermission(SecurityAction.Demand,
```

```
UnmanagedCode = true)]
```

```
unsafe public class Class1
```

```
{
```

```
private const int
```

```
WM_LBUTTONDOWNBLCLK
```

```
= 515;
```

```
private const int
```

```
WH_JOURNALRECORD
```

```
= 0;
```

```
public static int hHook = 0;
```

Re: QASetWindowsJournalHook by managed code

```
unsafe public static
EVENTMSG
myHookMessage = new
EVENTMSG();
unsafe public static
HookProc
MouseHookProcedure =
MouseHookProc;
public static IntPtr
pMouseHookProcedure =
Marshal.GetFunctionPointerForDelegate(MouseHookProcedure);

public static int
MouseHookProc(int nCode,
IntPtr wParam,
IntPtr lParam)
{
return
LibWrap.CallNextHookEx(hHook,
nCode, wParam,
lParam);
}

public static GCHandle gc1
=
GCHandle.Alloc(myHookMessage,
GCHandleType.Pinned);
public static GCHandle gc2
=
GCHandle.Alloc(MouseHookProcedure,
GCHandleType.Pinned);
public static GCHandle gc3
=
GCHandle.Alloc(pMouseHookProcedure,
GCHandleType.Pinned);

public static void
init_hook()
{
hHook =
LibWrap.QASetWindowsJournalHook(0,
pMouseHookProcedure,
gc1.AddrOfPinnedObject());
}
}

public class LibWrap
{
[DllImport("CoreDll.dll",
SetLastError = true)]
//public static extern int
```

Re: QASetWindowsJournalHook by managed code

```
QASetWindowsJournalHook(int
nFilterType, HookProc
pfnFilterProc, ref
EVENTMSG
pfnEventMsg);
public static extern int
QASetWindowsJournalHook(int
nFilterType, IntPtr
pfnFilterProc, IntPtr
pfnEventMsg);
[DllImport("CoreDll.dll",
EntryPoint =
"QAUnhookWindowsJournalHook",
SetLastError = true)]
internal static extern int
QAUnhookWindowsJournalHook(int
hook);
[DllImport("CoreDll.dll",
CharSet =
CharSet.Auto/*,CallingConvention
=
CallingConvention.StdCall*/)
public static extern int
CallNextHookEx(int
idHook, int nCode,
IntPtr wParam, IntPtr
lParam);
}
[StructLayout(LayoutKind.Sequential)]
public class EVENTMSG
{
public int message;
public Int16 x;
public Int16 y;

public int paramH;
public int time;
public int hwnd;
}
}
```

Appreciate,
JB.

"Chris Tacke, eMVP"
<ctacke.at.opennetcf.dot.com>
wrote in message
news:efslHVKtJHA.5836@xxxxxxxxxxxxxxxxxxxxxxxxxxxxx

Re: QASetWindowsJournalHook by managed code

You are still
using these
things
way
wrong. My
guess
(without
putting too
much grey
matter into
it) is that
you need to
create a
delegate
and call
GetFunctionPointForDelegate
and pass
that. You
also
shouldn't be
passing the
Target of
the
GCHandle
anywhere.

-Chris

"Jack P"

<private@xxxxxxxxxxxx>

wrote in
message

news:%23Yup05JtJHA.4452@xxxxxxxxxxxxxxxxxxxxxxxxxxxx

I
found
something
wrong
in
my
application,
in
fact
it
didn't
really
work,
why
I
didn't

Re: QASetWindowsJournalHook by managed code

see
crash
that
is
because
the
callback
didn't
assign
correctly
when
using
GCHandle.
I
list
my
complete
code
as
following,
it
packaged
as
C#
DLL,
it
will
crash
after
perform
any
other
program.
(
if
not
exit
this
program,
everything
looks
fine
)

using
System;
using
System.Collections.Generic;
using
System.Linq;
using

Re: QASetWindowsJournalHook by managed code

```
System.Text;
using
System.Runtime.InteropServices;
using
System.Security.Permissions;

namespace
ClassLibrary1
{
unsafe
public
delegate
int
HookProc(int
nCode,
IntPtr
wParam,
IntPtr
lParam);
[SecurityPermission(SecurityAction.Demand,
UnmanagedCode
=
true)]
unsafe
public
class
Class1
{
private
const
int
WM_LBUTTONDOWNBLCLK
=
515;
private
const
int
WH_JOURNALRECORD
=
0;
public
static
int
hHook
=
0;
unsafe
public
static
EVENTMSG
myHookMessage
```

Re: QASetWindowsJournalHook by managed code

```
=
new
EVENTMSG();
unsafe
public
static
HookProc
MouseHookProcedure
=
MouseHookProc;

unsafe
public
static
GCHandle
gc1
=
GCHandle.Alloc(myHookMessage,
GCHandleType.Pinned);
unsafe
public
static
GCHandle
gc2
=
GCHandle.Alloc(MouseHookProcedure,
GCHandleType.Pinned);

unsafe
public
static
int
MouseHookProc(int
nCode,
IntPtr
wParam,
IntPtr
lParam)
//////////
HERE
is
the
hookproc
//////////
{
return
LibWrap.CallNextHookEx(hHook,
nCode,
wParam,
lParam);
}
```

Re: QASetWindowsJournalHook by managed code

```
public
static
void
init_hook()
//////////
HERE
is
the
init
//////////
{
hHook
=
LibWrap.QASetWindowsJournalHook(0,(HookProc)
gc2.Target,
gc1.AddrOfPinnedObject());
}
}
```

```
public
class
LibWrap
//////////
class
for
DLL
//////////
{
[DllImport("CoreDll.dll",
 SetLastError
=
true)]
//public
static
extern
int
QASetWindowsJournalHook(int
nFilterType,
HookProc
pfnFilterProc,
ref
EVENTMSG
pfnEventMsg);
public
static
extern
int
QASetWindowsJournalHook(int
nFilterType,
HookProc
```

Re: QASetWindowsJournalHook by managed code

```
    pfnFilterProc,
    IntPtr
    pfnEventMsg);
[DllImport("CoreDll.dll",
EntryPoint
=
"QAUnhookWindowsJournalHook",
 SetLastError
=
true)]
internal
static
extern
int
QAUnhookWindowsJournalHook(int
hook);
[DllImport("CoreDll.dll",
 CharSet
=
CharSet.Auto/*,CallingConvention
=
CallingConvention.StdCall*/)
public
static
extern
int
CallNextHookEx(int
idHook,
int
nCode,
IntPtr
wParam,
IntPtr
lParam);
}
[StructLayout(LayoutKind.Sequential)]
public
class
EVENTMSG
{
public
int
message;
public
Int16
x;
public
Int16
y;

public
```

Re: QASetWindowsJournalHook by managed code

```
int  
paramH;  
public  
int  
time;  
public  
int  
hwnd;  
}  
}
```

Appreciate
for
your
help.
JB.

"Jack
P"
<private@xxxxxxxxxxxx>
wrote
in
message
news:OkQLro8sJHA.1492@xxxxxxxxxxxxxxxxxxxxxxxxxxxx

OK,
I
have
to
say
I
am
newbie
to
handle
something
:
D

Everything
great
now
:D

Appreciate!
Chris!

"Chris
Tacke,
eMVP"

Re: QASetWindowsJournalHook by managed code

got
passed
in
to
the
hook
procedure
(I
wouldn't
have
guessed
it),
but
the
address
is
the
slot
0
address.
When
you
change
programs,
your
app
is
no
longer
in
slot
0,
the
running
app
is,
and
so
that
slot
0
address
now
points
to
something
altogether
different,
and
when
the

Re: QASetWindowsJournalHook by managed code

hook
tries
to
call
it
it
pukes.

I
know
these
work
as
I've
done
them.

--

Chris
Tacke,
Embedded
MVP
OpenNETCF
Consulting
Giving
back
to
the
embedded
community
<http://community.OpenNETCF.com>

"Jack
P"
<private@xxxxxxxxxxxx>
wrote
in
message
<news:uH1pRA6sJHA.3928@xxxxxxxxxxxx>

Hi
Chris,
Thanks
for
help,
I
changed
code
as
follows

Re: QASetWindowsJournalHook by managed code

but
result
is
the
same,
regarding
the
leakage,
I
guess
you
mean
the
GCHandle
never
free
in
code,
I
guess
it
will
OK
because
enough
memory
for
leakage
...
just
to
make
sure
everything
should
exist
in
memory
all
the
time
....

public
class
Class1
{
public
static
int
hHook

Re: QASetWindowsJournalHook by managed code

```
=
0;
public
static
EVENTMSG
myHookMessage
=
new
EVENTMSG();
public
static
GCHandle
gc1
=
GCHandle.Alloc(myHookMessage,
GCHandleType.Pinned);
public
static
GCHandle
gc2
=
GCHandle.Alloc(MouseHookProced
GCHandleType.Pinned);
public
static
GCHandle
gc3
=
GCHandle.Alloc(hHook,
GCHandleType.Pinned);
...
public
static
void
init_hook()
{
myHookMessage.hwnd
=
0;
;
myHookMessage.message
=
0;
myHookMessage.paramH
=
0;
myHookMessage.time
=
0;
hHook
=
```

Re: QASetWindowsJournalHook by managed code

```
QASetWindowsJournalHook(0,  
MouseHookProcedure,  
ref  
myHookMessage);  
}
```

The
Class1
was
packaged
to
DLL
then
called
from
my
application,

I
am
not
sure
it
will
help
or
not.
or
I
should
package
Native
C
DLL
to
make
sure
no
memory
collection
?

I
still
can't
get
my
application
work
....
please
help,

Re: QASetWindowsJournalHook by managed code

appreciate!

J.B.

"Chris
Tacke,
eMVP"

<ctacke.at.opennetcf.dot.com>

wrote

in

message

news:eHeyiv4sJHA.4684@xxxxxxxx

My
guess
is
that
the
gchandle
is
going
out
of
scope
and
getting
collected.
If
not
you
still
have
a
memory
leak.

--

Chris
Tacke,
Embedded
MVP
OpenNETCF
Consulting
Giving
back
to
the
embedded
community

Re: QASetWindowsJournalHook by managed code

<http://community.OpenNETC>

"Jack
P"
<private@xxxxxxxxxxxx>
wrote
in
message
<news:%239cbS40sJHA.4632>

Hi
All,

I
am
trying
QASetWindowsJour
in
c#,
it
works
but
having
some
problem.

I
can
receive
the
hooked
message
correctly
(
mouse
move,
click,
x
,
y
...etc
)
),
but
as
long
as
I
do
something
as
below,

Re: QASetWindowsJournalHook by managed code

the
system
(ppc)
will
crash.

1.
Run
my
c#
QASetWindowsJour
application,
working.

2.
Click
[Start],
my
application
is
still
working,
message
received
well.

3.
Click
[Today]
or
[Setup]
or
[IE]

...
whatever
the
next
program
is

....
the
while
system
will
crash
(
stop
there
then
can't
do
anything
....

Re: QASetWindowsJournalHook by managed code

```
)  
  
public  
static  
int  
MouseHookProc(int  
nCode,  
IntPtr  
wParam,  
IntPtr  
lParam)  
{  
return  
CallNextHookEx(hF  
nCode,  
wParam,  
lParam);  
}
```

I
also
tried
to
GCHandle
everything
but
result
is
the
same
...

```
public  
static  
void  
init_hook()  
{  
EVENTMSG  
myHookMessage  
=  
new  
EVENTMSG();  
myHookMessage.me  
=  
0;  
myHookMessage.pa  
=  
0;  
myHookMessage.tim  
=  
0;
```

Re: QASetWindowsJournalHook by managed code

```
GCHandle
gc1
=
GCHandle.Alloc(my
GCHandleType.Pinn
GCHandle
gc2
=
GCHandle.Alloc(Mo
GCHandleType.Pinn
GCHandle
gc3
=
GCHandle.Alloc(hH
GCHandleType.Pinn
```

```
hHook
=
QASetWindowsJour
MouseHookProcedu
ref
myHookMessage);
}
```

and
I
also
tried
to
package
the
hook
functions
into
a
c#
DLL
...

but
result
is
the
same
...
will
also
crash
...

Re: QASetWindowsJournalHook by managed code

Can
somebody
help
?

Appreciate!
JB