

Re: Executing a method in a given thread context

Source:

<http://www.tech-archive.net/Archive/DotNet/microsoft.public.dotnet.framework.clr/2006-08/msg00100.html>

- *From:* "Francois PIETTE" <fpiette@xxxxxxxxxxxxxxxxxxxx>
 - *Date:* Mon, 14 Aug 2006 09:34:17 +0200
-

You might also look at the base class
System.Threading.SynchronizationContext.

I looked at MSDN for the description. It is not very clear to me how I use it.

Do you have any sample code, preferably using it from Socket.BeginReceive callback ? As you know, when the callback is run, it is in the context of a worker thread created by the system. From this callback, I want to raise an event in the context of the thread which called Socket.BeginReceive at the first place (not necessarily the mainthread).

—

Francois PIETTE
<http://www.overbyte.be>

"David Levine" <SnipHereDlevinenntp2AndHere@xxxxxxxxxx> a écrit dans le message de news:eHEFzgzvGHA.4384@xxxxxxxxxxxxxxxxxxxxxxxx

You might also look at the base class
System.Threading.SynchronizationContext. It provides some convenient abstractions for sending or posting messages between arbitrary threads.

The

nice part is that it does not care if the receiving thread has a window or not, it is the responsibility of the implementation of classes derived

from

SynchronizationContext to forward the message to the receiving thread.

Each

implementation of a SynchronizationContext must provide its own means of delivering the message within the correct synchronization context by

Re: Executing a method in a given thread context

providing its own implementation of the methods Send and Post. The default Post uses a threadpool thread for delivering the callback; for Send it invokes it directly (no synchronization).

The class System.Windows.Forms.WindowsFormsSynchronizationContext is derived

from this base class, and is used to synchronize messages sent to the application (it uses control.Invoke/BeginInvoke). The classes System.Windows.Forms.Application and System.Windows.Forms.Control both use this; internally it uses a Send or PostMessage to forward the message to

the

window

I used SynchronizationContext to provide a way for clients to subscribe to

a

generic event so that the client did not have to worry about synchronizing within the callback – when the client subscribes to the event it captures the caller's sync context and then uses that when it fires the event.

It seems to work well; it is only available in 2.0.

Dave

"Francois PIETTE" <fpiette@xxxxxxxxxxxxxxxxxxxx> wrote in message news:%23FsLCH4uGHA.1772@xxxxxxxxxxxxxxxxxxxxxxxxxxxx

Jeffrey,

Thanks for your very interesting answer. I will have a look at the

"timer

trick".

This raise some more questions:

1) What is the impact of relying on a winform control when my non-visual component is used in a non GUI application such as a console

application,

a

service application, a web application, or whatever ?

2) What is the impact of using P/Invoke ? I there anything to avoid ?

Should

Re: Executing a method in a given thread context

I use some attributes to mark the assemblies or the classes as safe and secure ?

3) Is the "UI thread" the main thread of an application or – as said by someone in a previous answer to my question – any thread which has a message

pump ? As I understand, it is the user interface thread aka the main thread.

4) If answer to 3 is "main thread", then how do I apply the "timer

trick"

to
a worker thread ?

--
Francois PIETTE
<http://www.overbyte.be>

""Jeffrey Tan[MSFT]"" <jetan@xxxxxxxxxxxxxxxxxxxxxx> a écrit dans le
message

de news:iScdys3uGHA.492@xxxxxxxxxxxxxxxxxxxxxxxxxxxxxx

Hi Francois,

Before I read this thread, you have several discussion with
the

community

members already. To not miss the key point, I want to recap
my
understand
of this issue.

Based on my understanding, you want to know some best
practise of
cross-thread operation in .Net winform and you do not like
the pure

Win32

p/invoke approach.

Yes, just as you have known, the only recommended way of
invoking UI

thread

Re: Executing a method in a given thread context

methods from other threads is using
Control.Invoke/BeginInvoke methods,

I

know you do not want to use them, but these 2 methods are
what .Net

Winform

provided for this purpose.

If you do not want the component user to take care of the
cross-thread
Control.Invoke/BeginInvoke requirement, you'd better refer
to
System.Timers.Timer class implementation model.
System.Timers.Timer

class

normally runs Elapsed event in another non-UI thread,
however, it
provided
a SynchronizingObject to deal with this. Once
SynchronizingObject is

set

to

certain Control reference, this Elapsed event will be executed
in that
control's UI thread without Timer class user aware of. This is
very
convenient for the class/component user and I think is what
you wanted

to

achieve. To achieve this, the Timer class uses
MyTimerCallback method

to

call the Elapsed event like this:

```
private void MyTimerCallback(object state)
{
....
try
```

Re: Executing a method in a given thread context

```
{
if (this.onIntervalElapsed != null)
{
if ((this.SynchronizingObject != null) &&
this.SynchronizingObject.InvokeRequired)
{

this.SynchronizingObject.BeginInvoke(this.onIntervalElapsed,
new
```

object[]

```
{

this, args1 });
}
else
{
this.onIntervalElapsed(this, args1);
}
}
}
catch (Exception)
{
}
...
}
```

Then in the Elapsed event, the user can write code does not care of the

UI

thread operation issue. You may use this design as well.

Additionally, there is rule regarding not use p/invoke.

Actually, as

you

have found out, .Net winform internally p/invoke Win32 API to complete must task. So if you want, you may use p/invoke freely. You may

encapsulate

all the Win32 API declaration in a class for good maintain ability.

Hope this helps.

Best regards,
Jeffrey Tan

Re: Executing a method in a given thread context

Microsoft Online Community Support

=====
Get notification to my posts through email? Please refer to

<http://msdn.microsoft.com/subscriptions/managednewsgroups/default.aspx#notif>

ications.

Note: The MSDN Managed Newsgroup support offering is for non-urgent issues where an initial response from the community or a Microsoft Support Engineer within 1 business day is acceptable. Please note that each follow up response may take approximately 2 business days as the support professional working with you may need further investigation to reach

the

most efficient resolution. The offering is not appropriate for

situations

that require urgent, real-time or phone-based interactions or complex project analysis and dump analysis issues. Issues of this nature are

best

handled working with a dedicated Microsoft Support Engineer by

contacting

Microsoft Customer Support Services (CSS) at
<http://msdn.microsoft.com/subscriptions/support/default.aspx>.

=====
This posting is provided "AS IS" with no warranties, and confers no

rights.

Re: Executing a method in a given thread context