

Forum:[Hack Forums](#)

Thread: [\[Tutorial\]Using threading instead of timer\[/Tutorial\]](#)

TLF-KoBE - November 10, 2010, 9:34 pm

You see it all the time, people saying "Dont use timers, they hog resources". Well i remember when i understood why not to, but never knew how not to. Here i will try and help people that do not know how, to learn how.

Here it goes.

First we are going to start of by creating a new sub routine. This is where the code that you would normally put in a timer is going to be.

```
Code Private sub action()  
messagebox.show("it worked")  
End Sub
```

Next lets declare some variables in the Form Class. Here you will see why we created the sub routine first.

```
Code Dim tTime As Threading.Timer  
Dim tCallback As Threading.TimerCallback = AddressOf action  
tCallback is the delegate that points to the method we want the timer to call. Which in our case is  
the sub we just created: action
```

Now all we have left to do is call the timer.

```
Code . Private Sub Button1_Click(ByVal sender As System.Object, ByVal e As  
System.EventArgs) Handles Button1.Click  
    tTime = New Threading.Timer(tCallback, nothing, 1000, 5000)  
End Sub
```

tCallback is our delegate that points to the sub action()
nothing is there because we are not passing any information to the sub. This could be used for
instance to send ByVal information to action()
1000 is setting the code to start after 1 second
5000 is setting the interval to repeat every 5 seconds.

Now to stop the timer, simply use this code:

```
tTime.Dispose
```

And that is the basic run down on an alternative to using timers.

I did not include how to access controls outside of this timer so as not to confuse people before they got a grasp on the basics. However it is quite simple. Once you have a grip on this and still cant figure it out, feel free to reply and i will show you the code. Thank you for reading.

Here is the full code. I have two buttons, Button1 starts it and Button 2 stops it.

```

Code Public Class Form1
    Dim tTime As Threading.Timer
    Dim tCallback As Threading.TimerCallback = AddressOf action

    Private Sub action(ByVal x As String)
        MessageBox.Show("it worked")
    End Sub
    Private Sub Button1_Click(ByVal sender As System.Object, ByVal e As System.EventArgs)
Handles Button1.Click
        tTime = New Threading.Timer(tCallback, Nothing, 1000, 5000)
    End Sub
    Private Sub Button2_Click(ByVal sender As System.Object, ByVal e As System.EventArgs)
Handles Button2.Click
        tTime.Dispose()
    End Sub
End Class

```

Elbobo - November 11, 2010, 4:37 am

uber fail. this isnt threading, its just creating a timer without actually having to drag the control onto the form.

Those people who say that timers hog resources know nothing, they just heard someone else say it and are constantly repeating it as their only knowledge
just cos its in the namespace threading doesnt mean its actually a thread.
heres how to declare a thread

```

Dim counter1 as new Count1()
Dim Thread1 as New System.Threading.Thread(Address of counter.Count)

```

```

Public Class Count1
'do shit
end sub

```

TLF-KoBE - November 11, 2010, 9:54 am

My apologies Elbobo, god of timers. I do not claim to be a master of timers or threading, however i got this from MSDN:

on System.Windows.Forms.Timer:

MSDN Wrote:This Windows timer is designed for a single-threaded environment where UI threads are used to perform processing. It requires that the user code have a UI message pump available and always operate from the same thread

and on System.Threading.Timer

MSDN Wrote:System.Threading.Timer is a simple, lightweight timer that uses callback methods and is served by thread pool threads. It is not recommended for use with Windows Forms, because its callbacks do not occur on the user interface thread.

To me, it sounds like its using threading. If anyone else thinks I suck at life and totally botched this Post, please let me know.. i'll remove the original post, shave my head and become a leech and only ask questions for the rest of my days. Oh, and i did try searching on why using timers hogged resources, and I came up empty. The only thing i could find is that it ties up the Form sometimes with bigger loads, and im assuming thats because its on the UI thread.

Somebody w/ some VB knowlege please shed some light for us.

Pulser - November 11, 2010, 5:08 pm

(Nov 11, 2010 05:37 AM)Elbobo Wrote: [\[CITE\]](#) uber fail. this isnt threading, its just creating a timer without actually having to drag the control onto the form.

Those people who say that timers hog resources know nothing, they just heard someone else say it and are constantly repeating it as their only knowledge just cos its in the namespace threading doesnt mean its actually a thread.
heres how to declare a thread

```
Dim counter1 as new Count1()  
Dim Thread1 as New System.Threading.Thread(Address of counter.Count)
```

```
Public Class Count1  
'do shit  
end sub  
You sir, are clearly Grandfather Time.
```

/sarcasm

Anyway, nice job genie.

bob42 - April 19, 2011, 8:53 am

Thank you for the share. This could come in handy for a codedom keylogger.

Doctor07 - April 19, 2011, 10:35 am

Very Nice & simple tutorial KoBE, good job.

idb. - April 19, 2011, 3:22 pm

I have written a simple threading tutorial that shows the multiple ways to do threading (Background Worker, Threadpool, Threading.Thread). If you guys are interested I would be more than happy to post it up. Let me know.

Ethereal - April 19, 2011, 3:32 pm

(Nov 11, 2010 05:37 AM)Elbobo Wrote: [\[CITE\]](#) uber fail. this isnt threading, its just creating a timer without actually having to drag the control onto the form.

Those people who say that timers hog resources know nothing, they just heard someone else say it and are constantly repeating it as their only knowledge just cos its in the namespace threading doesnt mean its actually a thread.
heres how to declare a thread

```
Dim counter1 as new Count1()  
Dim Thread1 as New System.Threading.Thread(Address of counter.Count)
```

```
Public Class Count1  
'do shit  
end sub  
There's more than one way to declare a thread, fuckwit.
```

Hell Guardian - April 19, 2011, 4:43 pm

Great tutorial KoBe Keep it up and ignore the trolls

BaussHacker - April 20, 2011, 2:17 am

This isreally inefficient. Learn about real threading, instead threading timers. Would be away more efficient :)

Ampix0 - April 20, 2011, 4:18 am

ok the deal here is, It is NOT an efficient timer. HOWEVER! It is like putting the timer control in its own thread. Which is still better. Not by a hell of a lot but if for some reason you needed dynamic timers or something this is good

TLF-KoBE - April 20, 2011, 9:11 pm

(Apr 20, 2011 02:17 AM)BaussHacker Wrote: [\[CITE\]](#) This isreally inefficient. Learn about real threading, instead threading timers. Would be away more efficient :)

Please explain 'real threading' to me... threading is threading. If the process is not running on a single thread, then logic would have most people assume that it is running multiple threads. I wrote this to show an alternative for using the timer control.

(Apr 20, 2011 04:18 AM)Ampix0 Wrote: [\[CITE\]](#) ok the deal here is, It is NOT an efficient timer. HOWEVER! It is like putting the timer control in its own thread. Which is still better. Not by a hell of a lot but if for some reason you needed dynamic timers or something this is good

not efficient by which means? please see my comment above and don't contradict yourself in your post next time.

????? - April 20, 2011, 9:19 pm

Well written tutorial very easy to understand will help members!

Alison Wonderland - April 20, 2011, 9:32 pm

Nice tut,

very usefull for people that get told to thread things yet get flamed for not knowing how.

Ethereal - April 21, 2011, 12:55 am

(Apr 20, 2011 04:18 AM)Ampix0 Wrote: [\[CITE\]](#) ok the deal here is, It is NOT an efficient timer. HOWEVER! It is like putting the timer control in its own thread. Which is still better. Not by a hell of a lot but if for some reason you needed dynamic timers or something this is good

How is it not efficient?

Please explain how you would be a timer in a thread, in an efficient manner. POC code is welcomed.

idb. - April 21, 2011, 1:01 am

What the OP did was not wrong per say... just unnecessary. If you want a threads operation to repeat over and over you can just use a Do or Loop instead of using a timer. There is no point in using the timer in this instance because it's only being used as an AddressOf pointer for the threads routine. Timers are used for continuous operation and keeping track of a timespan.

But yeah, there is no reason to flame the OP because his code WILL work and IS the proper code under certain circumstances.