

Forum:[Hack Forums](#)

Thread: [cross-thread, cross-class adding items to a listbox. \[RESOLVED WITH CODE\]](#)

Vrekt - July 15, 2015, 11:38 pm

UPDATE: Thanks everybody who replied and tried to help me, I learned some new things today :).

Here is how I solved the problem.

NOTE: all this code is done in the main form containing the listbox.

Thanks to Odus btw :P

Firstly in your other class containing the thread put

Code Public Shared flag as Object

Secondly the function to invoke the listbox:

Code Public Delegate Sub DelUpdateListBox(lb As ListBox, item As String)

Public Sub UpdateListBox(lb As ListBox, item As String)

If Me.InvokeRequired Then

Me.Invoke(New DelUpdateListBox(AddressOf UpdateListBox), New Object() {lb, item})

Else

lb.Items.Add(item)

End If

End Sub

^^ Does not have to be public I just did that because I'm a skid and thought I needed it...

Alright great! Now we need some variables..

Code ' // We are going to be doing this on another sub, so to stop it from writing the item multiple times we need this.

Private written As Boolean = False

' comparing the lastitem with the current to make sure there not the same.

Private lastline As String

Awesome. So now create a new sub named "additemtolistbox" (or whatever you prefer)

In this sub put this:

Code If yourotherclass.flag Is Nothing Then

Return

End If

If Not yourotherclass.flag Is Nothing And Not yourotherclass.flag Is lastline Then

lastline = CStr(yourotherclass.flag)

UpdateListBox(console, CStr(yourotherclass.flag))

End IfNOTE: Option Strict and Explicit is on.

Alright 2 more things now.

We basically want this sub to loop, so the lazy way to do so is call the sub in form paint event.

So now in your other thread in the other class, simply put

Code flag = "item to add"

That should give you a basic idea of how this would work. This code is probably buggy, but I hope it

helps some of you :)

tweety - July 16, 2015, 12:01 am

Saw thread title and walked in, then I saw the code.

Vrekt - July 16, 2015, 12:13 am

(Jul 16, 2015 12:01 AM)Tweety. Wrote: [\[CITE\]](#) Saw thread title and walked in, then I saw the code.
ahhaha yea I'm sort of a vb noob.

Vysse - July 16, 2015, 12:27 am

(Jul 15, 2015 11:38 PM)Vrekt Wrote: [\[CITE\]](#) Code Control.CheckForIllegalCrossThreadCalls = False
NO
NO
NONONO
NONONONONONO
NONONO
NO

SERIOUSLY JUST FUCKING NO
AND IF YOU THINK ABOUT IT AGAIN THEN NO. That is my input for this.

Shyon143 - July 16, 2015, 1:16 am

this is so new Imao idk what this is

I need help learning this would make more sense

Josh? - July 16, 2015, 1:23 am

Off the top of my head, in the form that has the listbox, create the following function:

```
Code Public Function addToListBox(ByVal LB As ListBox, ByVal item As Object) As Boolean
    Try
        If LB.InvokeRequired Then
            LB.Invoke(New MethodInvoker(Sub()
                LB.Items.Add(item)
            End Sub))
        Else
            LB.Items.Add(item)
        End If
        Return True
    Catch ex As Exception
        MsgBox("Error when adding item:" + vbCrLf + vbCrLf + ex.ToString(), vbExclamation +
vbOkOnly, "Error")
    End Try
    Return False
End Function
```

IntelliJ - July 16, 2015, 1:33 am

(Jul 15, 2015 11:38 PM)Vrekt Wrote: [\[CITE\]](#) Code ' // ignore this not really relevant

' // Check if notepad is running, if so decide what to do based on config.

' // Basic variable to tell the thread if the application is open, used for my loop here.

```
' // find the process

' // if its open

' // if a person wanted to close it when it started up execute this code

' // if they wanted to warn the user via a listbox on the main form.

' // wait 1ms before continuingFor reference:
VB comment: '
C# comment: //
There's no need to do both last time I checked.
```

HFTetramorph - July 16, 2015, 1:34 am

(Jul 16, 2015 01:33 AM)IntelliJ Wrote: [\[CITE\]](#) For reference:
VB comment: '
C# comment: //
There's no need to do both last time I checked.
User is assbat- But not wrong

IntelliJ - July 16, 2015, 1:34 am

(Jul 16, 2015 01:34 AM)BinaryEvolved Wrote: [\[CITE\]](#) User is assbat- But not wrong
Thanks sweetheart. ?

Vrekt - July 16, 2015, 1:35 am

(Jul 16, 2015 01:33 AM)IntelliJ Wrote: [\[CITE\]](#) For reference:
VB comment: '
C# comment: //
There's no need to do both last time I checked.
you are correct, just I prefer the look.

HFTetramorph - July 16, 2015, 1:36 am

Quote:Control.CheckForIllegalCrossThreadCalls = False

NO NO NO! JUST PROPERLY INVOKE THE NEXT THREAD! This is terrible practice and should never be used.

Vrekt - July 16, 2015, 1:50 am

(Jul 16, 2015 01:23 AM)Josh? Wrote: [\[CITE\]](#) Off the top of my head, in the form that has the listbox, create the following function:

```
Code Public Function addToListBox(ByVal LB As ListBox, ByVal item As Object) As Boolean
    Try
        If LB.InvokeRequired Then
            LB.Invoke(New MethodInvoker(Sub()
                LB.Items.Add(item)
            End Sub))
        Else
            LB.Items.Add(item)
        End If
        Return True
    Catch ex As Exception
        MsgBox("Error when adding item:" + vbCrLf + vbCrLf + ex.ToString(), vbExclamation +
```

```
vbOkOnly, "Error")
    End Try
    Return False
End Function
```

Nah didn't work, I must be doing something wrong, I have tried something similar to this before. I'm using this code to use the function.

NOTE: para_main is the form with the listbox.

Code para_main.addToListBox(para_main.console, "test")Thanks for the suggestion tho ?

Odus - July 16, 2015, 2:17 am

Delegate!

```
Code Private Delegate Sub DelUpdateListBox(lb As ListBox, item As String)
Private Sub UpdateListBox(lb As ListBox, item As String)
    If Me.InvokeRequired Then
        Me.Invoke(New DelUpdateListBox(AddressOf UpdateListBox), New Object() {lb, item})
    Else
        lb.Items.Add(item)
    End If
End Sub
```

Still not the most efficient way to do it though, I recommend the [ManagementEventWatcher](#) class.

Vrekt - July 16, 2015, 2:54 am

(Jul 16, 2015 02:17 AM)Odus Wrote: [\[CITE\]](#) Delegate!

```
Code Private Delegate Sub DelUpdateListBox(lb As ListBox, item As String)
Private Sub UpdateListBox(lb As ListBox, item As String)
    If Me.InvokeRequired Then
        Me.Invoke(New DelUpdateListBox(AddressOf UpdateListBox), New Object() {lb, item})
    Else
        lb.Items.Add(item)
    End If
End Sub
```

Thanks so much I implemented your code and it worked! Thanks everybody else as well.. I learned some new things today :). I updated my post with how I solved the problem, (very inefficient, but works) for anybody else that has the problem.

Vysse - July 16, 2015, 3:09 am

(Jul 16, 2015 02:54 AM)Vrekt Wrote: [\[CITE\]](#) Thanks so much I implemented your code and it worked! Thanks everybody else as well.. I learned some new things today :). I updated my post with how I solved the problem, (very inefficient, but works) for anybody else that has the problem.

Did you remove the CheckForIllegalCrossThreads = False? If not please do.

Vrekt - July 16, 2015, 3:11 am

(Jul 16, 2015 03:09 AM)Vysse Wrote: [\[CITE\]](#) Did you remove the CheckForIllegalCrossThreads = False? If not please do.

Yes I did, the only reason I added it was because some things didn't work, but I resolved them using better methods.

Vysse - July 16, 2015, 3:46 am

(Jul 16, 2015 03:11 AM)Vrekt Wrote: [\[CITE\]](#) Yes I did, the only reason I added it was because some things didn't work, but I resolved them using better methods.

Good to hear. Most people would take the easy way out and not learn from this.

+Haze - July 16, 2015, 5:24 am

Declaring a fully blown delegate has never been my thing. I prefer to use an anonymous lambda delegate sub.

```
Code Me.Invoke(Sub() listBox1.Items.Add(stuff))  
(Jul 16, 2015 02:17 AM)Odus Wrote: \[CITE\] Delegate!
```

```
Code Private Delegate Sub DelUpdateListBox(lb As ListBox, item As String)  
Private Sub UpdateListBox(lb As ListBox, item As String)  
    If Me.InvokeRequired Then  
        Me.Invoke(New DelUpdateListBox(AddressOf UpdateListBox), New Object() {lb, item})  
    Else  
        lb.Items.Add(item)  
    End If  
End Sub
```

Still not the most efficient way to do it though, I recommend the [ManagementEventWatcher](#) class. Wouldn't that just keep recursively calling it self if Invoke required is true?

Odus - July 16, 2015, 6:15 am

(Jul 16, 2015 05:24 AM)+Haze Wrote: [\[CITE\]](#) Wouldn't that just keep recursively calling it self if Invoke required is true?

Nope, because the delegate is invoking it (as a new event handler) when raised on another thread.

+Haze - July 16, 2015, 6:26 am

Ehh didn't know that. Learnt something new today. Haze.Knowledge++;