

Forum:[Hack Forums](#)

Thread: [\[Solution\] Form controls disposed errors with threading while closing abruptly](#)

Barret - August 13, 2017, 12:08 pm

At some point in time, if it didn't happen already, when you create forms and threads and it happens to be you close the form without closing the thread, any calls from within that thread that is making changes to let say, a textbox or some concept of a log view, it will flip you off with an exception.

Example exception

[https://msdn.microsoft.com/en-us/library/system.objectdisposedexception\(v=vs.110\).aspx](https://msdn.microsoft.com/en-us/library/system.objectdisposedexception(v=vs.110).aspx)

Controls and forms have properties such as "IsDisposed" and "IsAccessible" but neither of those do fuck all.

What we need is a proper locking mechanism that locks and block calls of such disposed controls. I saw a few methods, some even very elaborate using the OnClose procedures but yet they do not do what i want.

This might be an solution yet i did not fully test it but seems to work.

Quick setup:

Add the following to your form class

PHP Code

```
<?php
```

```
Private Property ExitLock As Object
```

```
Private Property ExitFlag As Boolean
```

And initialize them on OnFormLoad event

PHP Code

```
<?php
```

```
Me.ExitFlag = False
```

```
Me.ExitLock = New Object
```

And we add this to the OnFormClosing event

PHP Code

```
<?php
```

```
Me.FlagForShutdown()
```

Add this subroutine, this will be our enabler when we are closing so both parties are knowing "whats going on"

PHP Code

```
<?php
```

```
Private Sub FlagForShutdown()
```

```
    SyncLock Me.ExitLock
```

```

        Me.ExitFlag = True
    End SyncLock
End Sub

```

And in your thread worker, you make the following construction.
(Assuming you have setup a loop where a boolean determines the status of that loop)

PHP Code

```

<?php
Private Sub FlagForShutdown()
Synclock Me.ExitLock
    Do
        .....thread worker code....
        If (Me.ExitFlag) Then
            ...perform safe and clean exit code....
        End If
    While Bool
End Synclock
End Sub

```

And for the procedures that interact between the thread and form, you add the following.
(This is just an example of a output logger for some debug info coming from another thread)

PHP Code

```

<?php
Private Sub DebugLog(Message As String, Optional Clear As Boolean = False)
If (Not Me.ExitFlag) Then
    If (Me.InvokeRequired) Then
        Me.Invoke(Sub() Me.DebugLog(Message, Clear))
    Else
        If (Clear) Then Me.Textbox.Clear()
        Me.Textbox.AppendText(String.Format("{0}{1}", Message, Environment.NewLine))
        Me.Textbox.SelectionLength = Me.Textbox.Text.Length
        Me.Textbox.ScrollToCaret()
    End If
End If
End Sub

```

This will, in my idea, avoid having these exceptions. But i'm not entirely sure if this would be the best, so i'll leave this here and see who is going to add his post.

Ps. Yes i know of 'ManualResetEvent' but thats not what im asking here.

- B.

hfTP54 - August 20, 2017, 9:04 am

Wow man thanks so much I needed this as all my new themes rely on a global timer which can dispose objects so closing forms becomes a mess coz of this error. I'm surprised that no one else got this.

Rhez - August 20, 2017, 8:24 pm

Never encounter those error when working on thread. Maybe I can use this in the future.
Thanks Barret. ?

Barret - August 20, 2017, 11:38 pm

(Aug 20, 2017 08:24 PM)Rhez Wrote: [\[CITE\]](#) Never encounter those error when working on thread. Maybe I can use this in the future.

Thanks Barret. ?

Depends how you work your threads, if you have threads send off that do stuff and report back, while you happen to close the form itself, you will encounter this error "Trying to access a disposed control". So in order to prevent it, you must have a gate keeper that tells the callback to halt and return earlier without doing the reporting and die off.