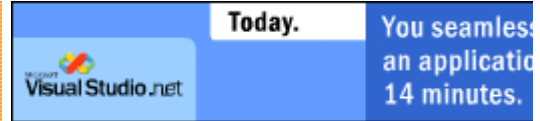


[DevX Home](#)[Premier Club](#)[Search](#)[Newsletters](#)[Skill Building](#)[Code Library](#)[Help](#)[Shop DevX](#)[Browse DevX](#)[Popular Topics](#)[Resource Centers](#)

- [.NET](#) ▪ [Java](#) ▪ [Web](#)
- [Database](#) ▪ [Open Source](#)
- [C++](#) ▪ [Visual Basic](#)
- [ASP](#) ▪ [UML](#) ▪ [XML](#)
- [Wireless](#)
- [All Discussions](#) >

[Featured Partners](#)

- [AMD64 DevSource](#)
- [BREW Wireless Resource Center](#)
- [Rainbow Technologies Security Center](#)
- [DevX Showcase for IBM developerWorks Toolbox](#)
- [Intel Optimizing Center](#)
- [DevX Skillbuilding from IBM developerWorks](#)
- [Nokia Mobile Development Channel](#)
- [IBM developerWorks Web Services Learning Center](#)
- [Destination .NET](#)
- [IBM Rational Developer PowerPacks](#)

[DevX Services](#)

[Premier Club](#)
Search over 1,600 technical books, access DevX premium content, *CoDe Magazine*, eLearning discounts, and more.

[White Paper Library](#)
Search thousands of white papers, case studies, webcasts and product documents.

Add Multithreading to Your VB.NET Applications (cont'd)

Synchronizing the Threads

VB.NET contains a few statements to provide synchronization of threads. In the Square example, you would want to synchronize the thread performing the calculation in order to wait for the calculation to complete so you can retrieve the result. Another example would be if you sort an array on a different thread and you would wait for that process to complete before using the array. To perform these synchronizations, VB.NET provides the SyncLock...End SyncLock statement and the Thread.Join method.

SyncLock gains an exclusive lock to an object reference that is passed to it. By gaining this exclusive lock you can ensure that multiple threads are not accessing shared data or that the code is executing on multiple threads. A convenient object to use in order to gain a lock is the System.Type object associated with each class. The System.Type object can be retrieved using the GetType method:


```
Public Sub CalcSquare()  
    SyncLock GetType(SquareClass)  
        Square = Value * Value  
    End SyncLock  
End Sub
```

Lastly, the Thread.Join method allows you to wait for a specific amount of time until a thread has completed. If the thread completes before the timeout that you specify, Thread.Join returns True, otherwise it returns False. In the square sample, if we did not want to raise events, we could call the Thread.Join method to determine if the calculation has finished. The code would look like the following:

```
Private Sub Button1_Click(ByVal sender As System.Object, _  
    ByVal e As System.EventArgs) Handles Button1.Click  
  
    Dim oSquare As New SquareClass()  
  
    t = New Thread(AddressOf oSquare.CalcSquare)  
  
    oSquare.Value = 30  
    t.Start()
```

W
s
i
d
c
t
h
m
t
h
a
r
s
e
a
i
n
a
g
e
v
p
l
f
o
m
u
r

Product Bank 
 Research components,
 tools, and enterprise
 solutions

DevX Marketplace 
 Shop our partner network
 for great selection and
 prices!

```

    If t.Join(500) Then
        MsgBox(oSquare.Square)
    End If
End Sub

```

The one thing to note with this method is that the procedure handling the event, in this case `SquareEventHandler`, will run within the thread that raised the event. It does not run within the thread from which the form is executing.

Matthew Arnheiter is a Senior Consultant at GoAmerica Communications (www.goamerica.net) of Hackensack, He has also the author of: The Visual Basic Developer's Guide to Design Patterns and UML (Sybex, 2000) He can be reached at (201) 996-1717 x2376 or at MArnheiter@goamerica.net.



Previous: Keep It Under Control

- 1** Introduction
- 2** Working with Threads
- 3** Keep It Under Control
- 4** Passing Data Through Multithreaded Procedures
- 5** Synchronizing the Threads

◀ Return to Get Help with Visual Basic

◀ Return to Main Get Help Page

Sponsored Links

[Make easy money with DellHost's FREE Client I](#)

[Windows interoperability just got easier! See how
 FREE TUTORIAL: Building Dynamic Web Sites](#)

[Create Collaborative Applications for your Enter
 for your FREE Enterprise Developer Kit Today!](#)

[Six Reasons Why Multithreading Wins on the D](#)