

Re: How is this conditional able to execute?

Source:

<http://www.tech-archive.net/Archive/DotNet/microsoft.public.dotnet.languages.vb/2005-04/msg03542.html>

- *From:* "Brett" <no@xxxxxxx>
 - *Date:* Tue, 19 Apr 2005 22:20:45 -0400
-

Thanks Stephany. This is super helpful.

Why is this part out of the loop:

```
Dim networkStream As NetworkStream = tcpClient.GetStream()
```

Also, your first IF:

```
If networkStream.DataAvailable then
```

didn't have an ending END IF. I put on just before the catch. Is that correct?

This conditional is still being executed when returndata = ""

```
If returndata <> "" AndAlso returndata <>
```

```
ClientForm.ReturnData Then
```

```
ClientForm.ReturnData = returndata
```

```
End If
```

Maybe I just don't understand the returndata = "" because of the byte array. But it seems you are testing the same conditions I did before. So the above will always execute because the array always has 8192 length and somehow that makes returndata pass on returndata <> "".

How can I read the string in :

```
Dim returndata As String = Encoding.UTF8.GetString(bytes, 0, bytesread)
```

returndata gets the empty string here, although it really isn't, as have seen previously.

thanks,
brett

"Stephany Young" <noone@localhost> wrote in message

<news:eJau0ZURFHA.576@xxxxxxxxxxxxxxxxxxxxxxxxxxxx>

> The mist is clearing.

>

> If I've misunderstood anything please let me know because I'm still making
> some assumptions.

>

> I suspect that your starting point for the Try ... Catch ... End Try

> section is the example shown in the documentation on the

Re: How is this conditional able to execute?

- > TcpClient.GetStream method.
- >
- > What you have done is wrapped it in a loop and added some embellishments
- > and that example was not quite designed for that to be done to it. The
- > example was designed for a one-time run and to drop through the logic to
- > whatever followed. In your case you want the loop to continue until
- > certain conditions are met or the program ends. To meet this need you need
- > to do things just a little differently.
- >
- > I assume that ClientForm.utility is also a shared reference to some sort
- > of utility class and that ClientForm.Packet is declared in a similar
- > fashion to ClientForm.ReturnData.
- >
- > If I am correct in that ConnectInit is outside the thread then you
- > shouldn't be attempting to reinstate a 'broken' connection within the
- > thread. If a 'broken' connection is detected you should allow the thread
- > to 'die', then reconnect, then start a new instance of the thread.
- >
- > Before the start of loop you are obtaining a reference to the
- > NetworkStream object of the TcpClient object. In addition, on each
- > iteration of the loop you are obtaining a fresh reference to the
- > NetworkStream object of the TcpClient object. There is no reason to do
- > this. Don't quote me on this but I suspect that doing so will cause you
- > problems.
- >
- > The declaration of the 'bytes' array should not need to be inside the
- > loop. Creating a new array on every iteration is not necessary. When you
- > read the NetworkStream object, the data will overwrite what is already in
- > the array. of course the array should be created with 8192 elements by
- > using tcpClient.ReceiveBufferSize - 1 rather than the 8193 elements you
- > will get if you simply use tcpClient.ReceiveBufferSize.
- >
- > The NetworkStream.Read method, as well as filling the array, also returns
- > an integer indicating the number of bytes successfully read into the
- > array. You should make use of this to control what you are dealing with.
- > Remember that tcpClient.ReceiveBufferSize is the maximum number of bytes
- > that can be read at any one time rather than the number of bytes that are
- > actually read.
- >
- > For the third parameter to the NetworkStream.Read method, it is, in my
- > opinion, better to use bytes.Length rather than
- > tcpClient.ReceiveBufferSize because bytes.Length is the actual number of
- > elements in the array.
- >
- > I think that the tests of the NetworkStream.CanWrite and
- > NetworkStream.CanRead are probably redundant, unless you are opening the
- > sockets at either end with specific read/write restrictions.
- >
- > Another important point is that the NetworkStream.Read method will block
- > until data is available. This is probably undesirable but can be
- > circumvented by using the NetworkStream.DataAvailable property. This will

Re: How is this conditional able to execute?

> allow your loop to iterate in a timely manner.
>
> Now that we know exactly how many bytes were read from the NetworkStream
> object, we can use the appropriate Encoding.GetString overload to make
> sure that our string is a representation of only those bytes. This should
> do away with the need for any 'trimming' of the resulting string.
>
> Where you have the test for NetworkStream.DataAvailable is inappropriate
> and I get the impression that you might not fully understand it's use.
>
> On the test of returndata, the .ToString is redundant because returndata
> is already a string. The parenthesis are also redundant. In addition,
> using AndAlso will be better than And here because if returndata is an
> empty string then the second part of the test will not be evaluated if
> AndAlso is used whereas it will be evaluated if just And is used. The
> logic is that if the evaluation of first part fails then why bother
> evaluating the second part at all. This is called 'short-circuiting' and is
> a very useful technique when used appropriately. Setting returndata to
> Nothing is redundant because returndata is going to be created anew on the
> next iteration of the loop. Likewise, the 'Else' that does nothing is also
> redundant.
>
> As you can see, the resulting code is more compact and now, that the
> appropriate techniques are being used, should work quite happily.
>
> What is left to do, of course, is for you to add the appropriate code to
> 'tear-down' the thread in a friendly fashion when the program needs to
> terminate or you need to stop the thread for some other reason.
>
>
> Code as supplied by Brett
> -----
>
> In ClientForm
> -----
> Public Shared ReturnData As String
> Public Shared Packet As String ' or something like that
> Public Shared utility As UtilityClass = New UtilityClass ' or something
> like that
> -----
>
>
> Dim ConnectAttempts As Integer = 0
> Dim t As Thread
>
> Dim Packet As String
> Dim InitialIDSent As Boolean = False
>
> Dim tsGap As TimeSpan
> Dim dtStart As DateTime = Now
>

Re: How is this conditional able to execute?

Re: How is this conditional able to execute?

```
> ConnectInit()
>
> Within thread
> -----
> Dim networkStream As NetworkStream = tcpClient.GetStream()
>
> While True
> Try
>
> If networkStream.CanWrite And networkStream.CanRead Then
>
> networkStream = tcpClient.GetStream()
>
> Dim bytes(tcpClient.ReceiveBufferSize) As Byte
> networkStream.Read(bytes, 0, tcpClient.ReceiveBufferSize)
> Dim returndata As String =
> Trim(Encoding.UTF8.GetString(bytes).Trim(Char.MinValue))
>
> If InitialIDSent Then
> Packet = "Nothing to Write from: " & Thread.CurrentThread.Name
> Else
> Packet = "Connected: " & ClientForm.Packet
> InitialIDSent = True
> 'write to log file
>
> ClientForm.utility.WriteToFile(ClientForm.CurrentDirectory.GetCurrentDirectory,
> ClientForm.LogFileExec, ClientForm.utility.CurrentDateTime(0) & " –
> Connected: " & ClientForm.Packet)
> End If
>
> ' Do a simple write.
> Dim sendBytes As [Byte]() = Encoding.ASCII.GetBytes(Packet.TrimEnd)
> networkStream.Write(sendBytes, 0, sendBytes.Length)
>
> If networkStream.DataAvailable() Then
> If (returndata <> "") And returndata.ToString <>
> ClientForm.ReturnData Then
> ClientForm.ReturnData = returndata
> returndata = Nothing
>
> Else
> End If
> End If
>
> End If
>
> Catch ex As Exception
>
> If ex.Message = "Operation not allowed on non-connected sockets."
> Then
> ConnectInit()
```

Re: How is this conditional able to execute?

Re: How is this conditional able to execute?

```
> ConnectAttempts = +1
>
> If ConnectAttempts > 5 Then
> Exit Sub
> End If
> Else
> End If
> Exit While
> End Try
>
> Thread.CurrentThread.Sleep(500)
> End While
> -----
>
>
> Code as modified by Stephany
> -----
>
> In ClientForm
> -----
> Public Shared ReturnData As String
> Public Shared Packet As String ' or something like that
> Public Shared utility As UtilityClass = New UtilityClass ' or something
> like that
> -----
>
>
> Dim ConnectAttempts As Integer = 0
> Dim t As Thread
>
> Dim Packet As String
> Dim InitialIDSent As Boolean = False
>
> Dim tsGap As TimeSpan
> Dim dtStart As DateTime = Now
>
> ConnectInit()
>
> Within thread
> -----
> Dim networkStream As NetworkStream = tcpClient.GetStream()
>
> Dim bytes(tcpClient.ReceiveBufferSize - 1) As Byte
>
> Dim bytesread As Integer
>
> While True
> Try
> If networkStream.DataAvailable then
> bytesread = networkStream.Read(bytes, 0, bytes.Length)
> Dim returndata As String = Encoding.UTF8.GetString(bytes, 0,
```

Re: How is this conditional able to execute?

Re: How is this conditional able to execute?

```
> bytesread)
> If InitialIDSent Then
> Packet = "Nothing to Write from: " & Thread.CurrentThread.Name
> Else
> Packet = "Connected: " & ClientForm.Packet
> InitialIDSent = True
> 'write to log file
>
> ClientForm.utility.WriteToFile(ClientForm.CurrentDirectory.GetCurrentDirectory,
> ClientForm.LogFileExec, ClientForm.utility.CurrentDateTime(0) & " -
> Connected: " & ClientForm.Packet)
> End If
> ' Do a simple write.
> Dim sendBytes() As Byte = Encoding.ASCII.GetBytes(Packet.TrimEnd)
> networkStream.Write(sendBytes, 0, sendBytes.Length)
> If returndata <> "" AndAlso returndata <> ClientForm.ReturnData
> Then ClientForm.ReturnData = returndata
> End If
> Catch ex As Exception
> Exit While
> End Try
> Thread.CurrentThread.Sleep(500)
> End While
> -----
>
>
>
```

• **Follow-Ups:**

- ◆ **Re: How is this conditional able to execute?**
◇ From: Stephany Young

• **References:**

- ◆ **How is this conditional able to execute?**
◇ From: Brett
- ◆ **Re: How is this conditional able to execute?**
◇ From: Stephany Young
- ◆ **Re: How is this conditional able to execute?**
◇ From: Brett
- ◆ **Re: How is this conditional able to execute?**
◇ From: Stephany Young
- ◆ **Re: How is this conditional able to execute?**
◇ From: Brett
- ◆ **Re: How is this conditional able to execute?**
◇ From: Stephany Young
- ◆ **Re: How is this conditional able to execute?**

Re: How is this conditional able to execute?

Re: How is this conditional able to execute?

◇ *From:* Brett

◆ **Re: How is this conditional able to execute?**

◇ *From:* Stephany Young

◆ **Re: How is this conditional able to execute?**

◇ *From:* Brett

◆ **Re: How is this conditional able to execute?**

◇ *From:* Stephany Young

◆ **Re: How is this conditional able to execute?**

◇ *From:* Brett

◆ **Re: How is this conditional able to execute?**

◇ *From:* Stephany Young

◆ **Re: How is this conditional able to execute?**

◇ *From:* Brett

◆ **Re: How is this conditional able to execute?**

◇ *From:* Stephany Young

◆ **Re: How is this conditional able to execute?**

◇ *From:* Brett

◆ **Re: How is this conditional able to execute?**

◇ *From:* Stephany Young

◆ **Re: How is this conditional able to execute?**

◇ *From:* Brett

◆ **Re: How is this conditional able to execute?**

◇ *From:* Stephany Young

◆ **Re: How is this conditional able to execute?**

◇ *From:* Brett

◆ **Re: How is this conditional able to execute?**

◇ *From:* Stephany Young

- Prev by Date: **Re: Code needs to be alerted that a removable disk drive's state**
- Next by Date: **RE: Access HKCU from an app with another set of credentials**
- Previous by thread: **Re: How is this conditional able to execute?**
- Next by thread: **Re: How is this conditional able to execute?**
- Index(es):
 - ◆ **Date**
 - ◆ **Thread**