

## Re: Ensure single class instance but with multiple references to it

---

*Source:*

<http://www.tech-archive.net/Archive/Excel/microsoft.public.excel.programming/2006-01/msg00665.html>

---

- *From:* [hooksie2@xxxxxxxxxxxx](mailto:hooksie2@xxxxxxxxxxxx)
  - *Date:* 4 Jan 2006 17:58:24 -0800
- 

Thanks for the reply – it certainly gave me some things to think about.

At the moment I instantiate the activeX object in the class and store the reference there too, ie. cOpenServer class looks like:

```
Private m_objOpenServer as Object
```

```
Private Sub Class_Initialize()  
If m_objOpenServer Is Nothing Then  
Set m_objOpenServer = CreateObject("PX32.OpenServer.1")  
End If  
End Sub
```

Applying your suggestion directly would mean that only the first instance of the class actually works. Subsequent instances won't have an ActiveX reference so the rest of the functions won't work.

I guess one option would be to make m\_objOpenServer a global variable in a standard module and if OpenServerUserCount > 1 then just set a new reference to the global variable. Only thing I don't like about this though is that now all the activeX stuff isn't encapsulated in my class object which was the original intent.

The other option I see would be to apply your suggestion in my original GetOpenServer and TerminateOpenServer module functions, ie.

```
Private m_clsOpenServer as cOpenServer  
Public g_lngOpenServerObjCount As Long  
  
Public Function GetOpenServer() As cOpenServer  
If m_clsOpenServer Is Nothing Then  
g_lngOpenServerObjCount = 1  
Set m_clsOpenServer = New cOpenServer  
Else  
g_lngOpenServerObjCount = g_lngOpenServerObjCount + 1  
End If  
Set GetOpenServer = m_clsOpenServer
```

Re: Ensure single class instance but with multiple references to it

End Function

```
Public Sub TerminateOpenServer()  
g_lngOpenServerObjCount = g_lngOpenServerObjCount - 1  
If g_lngOpenServerObjCount <= 0 Then  
Set m_clsOpenServer = Nothing  
End If  
End Sub
```

This seems pretty dangerous though since TerminateOpenServer could easily be called when a real instance hadn't actually been created (ie. as part of tidy up code after an error).

I'm still a bit surprised that m\_clsOpenServer can be destroyed even when another variable has a reference to my object through it but I guess that's just my bad luck.

Please let me know if I've misunderstood anything or if there is another approach that might work.

Thanks again,  
Andrew

Tushar Mehta wrote:

```
> You need to use variables that are shared by all instances of the  
> class. Unfortunately, VBA doesn't support that concept directly --  
> well, at least as far as I have been able to figure out, it doesn't. A  
> workaround, while easy, leaves you vulnerable to (future) lazy  
> programmers.  
>  
> Create a new *standard* module called cOpenServerGlobals. This will  
> hold the globals for your cOpenServer class. This module will contain  
>  
> Option Explicit  
> Option Private Module  
> Public OpenServerUserCount As Long  
> Public x 'reference to the activex component  
>  
> In the cOpenServer class module, add Initialize and Terminate  
> procedures:  
> Option Explicit  
>  
> Private Sub Class_Initialize()  
> OpenServerUserCount = OpenServerUserCount + 1  
> If OpenServerUserCount = 1 Then  
> 'instantiate x  
> End If  
> End Sub  
>  
> Private Sub Class_Terminate()
```

Re: Ensure single class instance but with multiple references to it

Re: Ensure single class instance but with multiple references to it

```
> OpenServerUserCount = OpenServerUserCount - 1
> If OpenServerUserCount = 0 Then
> 'close and release x
> End If
> End Sub
>
> Now, you can use the cOpenServer class like any other class module. It
> will correctly handle the linking to the ActiveX component.
>
> Option Explicit
>
> Dim aServerRef As cOpenServer
> Sub doSomething()
> Set aServerRef = New cOpenServer
> '...
> Set aServerRef = Nothing
> End Sub
> or
>
> Option Explicit
>
> Sub doSomething()
> Dim aServerRef As cOpenServer
> Set aServerRef = New cOpenServer
> '...
> End Sub
>
> --
> Regards,
>
> Tushar Mehta
> www.tushar-mehta.com
> Excel, PowerPoint, and VBA add-ins, tutorials
> Custom MS Office productivity solutions
>
> In article <1136361625.601941.165970@xxxxxxxxxxxxxxxxxxxxxxxxxxxxxx>,
> hooksie2@xxxxxxxxxxx says...
>> I have written a class module which creates a link to an ActiveX server
>> and provides some functions. Because the server doesn't seem to be
>> able to handle multiple references being set to it I am trying to
>> ensure that only one instance of my class is created and that all other
>> attempts to create a new instance merely set a pointer to the existing
>> instance. To do this I have the following in a standard module:
>>
>> Private m_clsOpenServer As cOpenServer
>>
>> Public Function GetOpenServer() As cOpenServer
>> If m_clsOpenServer Is Nothing Then
>> Set m_clsOpenServer = New cOpenServer
>> End If
>> Set GetOpenServer = m_clsOpenServer
```

Re: Ensure single class instance but with multiple references to it

```
>> End Function
>>
>>
>> I then reference the class by:
>> Dim clsOpenServer As cOpenServer
>> Set clsOpenServer = GetOpenServer
>>
>> When I'm finished I destroy the object, ie.
>> Set clsOpenServer = Nothing
>>
>> Now comes the problem. Even though clsOpenServer is destroyed,
>> m_clsOpenServer remains in memory and so there is still a reference to
>> my class object and to the ActiveX server.
>>
>>
>> I thought the thing to do here was to add a terminate procedure:
>> Public Sub TerminateOpenServer()
>> Set m_clsOpenServer = Nothing
>> End Sub
>> and then call this after clsOpenServer is set to nothing. If there
>> were other pointers set to m_clsOpenServer then I thought it would be
>> held in memory even though I set it to nothing. This is not the case
>> however (not sure why not??) and so the next time I call GetOpenServer
>> a new instance is created (and then pretty soon the application I am
>> calling freezes up).
>>
>> Is there any robust way to handle this type of situation?
>>
>> Thanks a lot,
>> Andrew
>>
>>
```

---

• *Follow-Ups:*

- ◆ **Re: Ensure single class instance but with multiple references to it**  
◇ From: Tushar Mehta

• *References:*

- ◆ **Ensure single class instance but with multiple references to it**  
◇ From: hooksie2
- ◆ **Re: Ensure single class instance but with multiple references to it**  
◇ From: Tushar Mehta

- Prev by Date: **Re: Project Unviewable???**
- Next by Date: **Re: range average code**
- Previous by thread: **Re: Ensure single class instance but with multiple references to it**
- Next by thread: **Re: Ensure single class instance but with multiple references to it**

Re: Ensure single class instance but with multiple references to it

- Index(es):
  - ◆ *Date*
  - ◆ *Thread*