

Re: Problem with asp.net app only allowing 1 user at a time

Source:

<http://www.tech-archive.net/Archive/DotNet/microsoft.public.dotnet.framework.aspnet/2004-12/1838.html>

From: John M Deal (*johndeal_at_necessitysoftware.com*)

Date: 12/07/04

Date: Tue, 07 Dec 2004 07:30:36 -0800

I really wish I had more time to get into this in depth, but I'm at work right now so I'll have to keep this short and rely on your ability to run with this info.

In ASP.Net (and classic ASP for that matter) there are a number of intrinsic/built-in object provided globally by the environment itself. Included in these objects is Response (the object that can send data back to the client), Request (the object that contains the data sent from the client to the server), Application (an object that can store globally available data), and Session (an object that can store user specific data). There are others but these will get you started. You do not need to create these objects, you simply need to refer to them in your code.

Now in your case you are reading and writing everything to the database, so my original notes may not apply in your situation, you'll have to decide that. However if you do decide to try and follow what I said before you should know a couple of things.

First Application is created when the web application first starts up and is available until the web application shuts down. You can catch events that tell you when these happen in the Global.asax file(s). You'll want to look into this because if you don't prepare your code for it data may disappear during long periods of user inactivity and you may want to reload it when the application reloads. The Application object can be accessed by any of the connections/threads that are currently active on the server. This means that you need to lock the access to the Application object when modifying its contents so you don't run into threading problems; to do this use Application.Lock() before you add something to it and Application.Unlock() after you are done with it. Any page that works with the Application object will be using data that is available to all users connected to the system.

Second, there is the Session object. By default session only lasts twenty minutes from the last time a user accessed the system (the server

does not have any other way to know if the user is still connected other than if they make a request). This time limitation can be controlled in the web.config file for the site. Session is only valid for a single user connected to the system, however be aware that if the user opens two separate browsers each one will have its own session (unless one was launched from another then they share the same session). Session uses a cookie by default to track an unique id that is used to retrieve data for the specific user that it is associated to. When you store object instances in Session they are only stored for (and retrievable by) the browser instance associated to the unique id. Session can be made to not require cookies (but then it puts the unique id in the querystring for the page). Also Session can be configured to store its data in memory (InProc) in an out of process Service (StateServer) or in an external database (SqlServer). This is controlled by your web.config file for the website (as is the session timeout). If you choose to store session anywhere besides InProc (the default) you will have to make sure any object that you store in it is serializable or the system will throw an exception (it uses serialization to marshal the object across the process boundary between the site and the storage repository).

Now that all being said. Whether you use Application or Session you store instances of objects (or primitives if you want) using a name/value methodology. For example if you have an instance of an User object called myUser and you want to put it in session you would do something like (C# code):

```
Session["TheUserObjectKey"] = myUser;
```

To retrieve this you would do something like:

```
User myUser = (User) Session["TheUserObjectKey"];
```

This would work the same for the Application though you'd want to do the Lock/Unlock pattern that I mentioned before. Also note that because Application and Session store the base Object type you have to cast your object back to the appropriate type when retrieving them back out.

Ok, that ran a bit longer than I thought it would but it is the basics of how these objects work. Now if you want to store objects for the users, you will want to see if their Session has an instance of your data object, if it doesn't you'd query it from the database then put it in Session, if it does you'll want to retrieve it from session and display the appropriate values. Obviously you'll have to deal with the issues involved with updating these data objects (if that is necessary for your circumstances) and also you must realize that because Application and Session have timeouts you will have objects hanging out in memory until they expire (and then until the GC collects the object) but this is standard for ASP.Net applications.

Hopefully this is enough to get you started. See the help files and MSDN online documentation for more information. If you are looking to

do enterprise development in ASP.Net I'd really recommend that you pickup a good ASP.Net book (search this newsgroup I'm sure you'll find plenty of recommendations). Also there are some webcasts being done on ASP.Net out on Microsoft's site that might give you some pointers. Hope something here helps.

Have A Better One!

John M Deal, MCP
Necessity Software

jsale wrote:

> I'm kinda new to professional asp.net development, so could you just clarify
> the application and session objects?
> I've got a module that houses many functions that all retrieve information
> about their specific table from the database, writing the information into a
> '.vb' class file. In effect, i am taking a snapshot of the database at that
> current moment and storing it to the class, amending it there and then
> writing changes back to the database afterwards. If i understand you
> correctly, that is application level, therefore every user's instance of the
> application sees the same data, whereas on a session level, each user would
> see their own instance. Is this correct? If so, do you have any pointers
> about making class files per session, not per application? I'm hugely
> confused! The reason i'm developing like this is because i'm migrating an old
> VB6 app and web-enabling it using ASP.NET. The old version used classes and i
> was expected to re-use as much as possible. I think i understand that the
> inherent difference is that with the old app it was installed seperately on
> each computer, whereas the ASP.NET version is just one copy, installed on the
> web server.
> Any help would be gratefully received :)
>
> "John M Deal" wrote:
>
>
>>If I'm understanding you correctly you are loading data for the user
>>from the database then storing that data into application scope. The
>>issue that you then have is that you retrieve this data from application
>>scope the next time the page is displayed. If this is the case then you
>>should modify the way you are handling state management.
>>
>>The Application object should only be used to store data that is truly
>>global to every user of the application. If you need to store things on
>>a per user basis you should look into using the Session object. Using
>>the Session object will allow you to store data classes on a per user
>>basis.
>>
>>If you are worried about running the same query against the database
>>multiple times, then you might want to abstract the use of the
>>Application object (not a bad idea in general anyway) so that you can
>>store a collection of query result sets keyed by the query used, then
>>retrieve the appropriate result set for each user from the collection.

microsoft.public.dotnet.framework.aspnet: Re: Problem with asp.net app only allowing 1 user at a time

>>If the collection doesn't contain the appropriate result set then you
>>query the database and then add it to the collection and put the
>>collection back in Application scope. Once you retrieve the appropriate
>>result set from the application collection you could then put it in the
>>user's Session object and continue your retrieval from there.
>>
>>Without more information on how you really intend to use these objects
>>it is hard to know which solution to use, but hopefully something from
>>the above will work for you.
>>
>>Have A Better One!
>>
>>John M Deal, MCP
>>Necessity Software
>>
>>jsale wrote:
>>
>>>I'm currently using ASP.NET with VS2003 and SQL Server 2003. The ASP.NET app
>>>i have made is running on IIS v6 and consists of a number of pages that allow
>>>the user to read information from the database into classes, which are used
>>>throughout the application. I have made class collections which, upon reading
>>>from the DB, create an instance of the class and store the DB values in
>>>there temporarily. My problem is that if user1 looks at record1, then user2
>>>looks at record2, anything user1 does from that point onwards points the data
>>>at record2.
>>>Is there anything in particular i need to do to make the asp.net application
>>>available to multiple users at the same time, each looking at different
>>>records in the DB?
>>>Thanks in advance
>>