

Re: Garbage Collection Issues in long-standing services

Source:

<http://www.tech-archive.net/Archive/DotNet/microsoft.public.dotnet.languages.csharp/2005-12/msg04500.html>

- *From:* Larry Herbinaux <LarryHerbinaux@xxxxxxxxxxxxxxxxxxxxxxxxxxxxxxxx>
 - *Date:* Thu, 22 Dec 2005 11:38:03 -0800
-

Willy, thanks for pointing this out, it does make sense concerning the Pinned array.

In our case, the client should be fulfilling the requests in an immediate fashion. If this is the case, the pinned array should get unpinned or compacted since the async call completes pretty quickly, right?

The receive case is done quite well, I do create a 4K buffer and reuse it for the life of the connection. In most cases, I don't expect an outstanding begin receive at the end of the connection, but it could happen. I do keep track of the IAsyncResult and if it is outstanding, I grab the state object and remove the reference to my wrapper socket class, but I don't remove the reference to the buffer. I should probably do this, right? The OS still has a reference to the request object, can I just call EndReceive if it is outstanding (not completed)? The one issue I see with this is if for some reason the remote (client) socket hasn't closed (e.g. physical route is broken, etc.), then EndReceive would block according to MSDN (framework 1.1).

The send case is a little more difficult and I would appreciate your advise on it. I also have to support SSL and there is a maximum limit of 16K that can be transferred at a time. I try to make life easy on the application programmer by buffering this data if the amount is larger than 16K. I have a class that my wrapper socket class references that contains an ArrayList of byte arrays, so there could be more than one. Should I go to the extreme of having a permanent send buffer and then just copy the data to this prior to sending so that OS has only one reference to a byte array during the length of the connection?

One other note, sorry about the long set of questions, I did use the CLR Profiler using a simple application that mimics the references of my TCPServer. My one worry was that my wrapper socket class did have a reference to the TCPServer object (which is obviously long-standing) and I was worried about the GC not collecting due to this. The TCPServer also has a hashtable that references the wrapper socket object. I did verify that

w

- Next by thread: [***Re: Garbage Collection Issues in long-standing services***](#)
- Index(es):

Re: Garbage Collection Issues in long-standing services

- *Date*
- *Thread*