

Re: Multithreading – writing to same file C#

Source:

<http://www.tech-archive.net/Archive/DotNet/microsoft.public.dotnet.languages.csharp/2005-09/msg00971.html>

- *From:* "Ignacio Machin \(.NET/ C# MVP \)" <ignacio.machin AT dot.state.fl.us>
 - *Date:* Wed, 7 Sep 2005 08:53:19 -0400
-

Hi,

Use Lambert's suggestion to avoid the main thread to end,
Also I would change your code,

```
public class yyy
{

private object o = new object();
StreamWriter sw = File.AppendText(@"d:\text.txt") ;

public static void Main()
{
Thread t;
for(int i=1;i<=3;i++)
{
t = new Thread(new ThreadStart(new zzz().abc));
t.Start();

}
Console.ReadLine() ;
}

public void abc()
{
//Calls to DAL methods

lock(o)
{
sw.WriteLine( .... );
}
}

}
```

cheers,

—

Ignacio Machin,

Re: Multithreading – writing to same file C#

ignacio.machin AT dot.state.fl.us
Florida Department Of Transportation

<mahendranepali@xxxxxxxx> wrote in message
news:1126078025.950266.293010@xx

>I am making a simulator application that will simulate different
> concurrent users connecting to a database. the application creates
> different userkeys and pushes them to the database tables. The business
> procedures hence interpret the userkey as a seperate request and act
> accordingly. For this I need to check the response time to the DB for
> say thousands of concurrent users. So what I did created Threads in a
> Loop and each thread invoked the DAL Class to query the DB. I just had
> to log the response (time to return dataset) of the DAL methods and log
> them to a text file.
> this is a sample code for multithreads writing to the file.

```
>
> public class yyy
> {
>
> private Mutex mut = new Mutex();
>
> public static void Main()
> {
> Thread t;
> for(int i=1;i<=3;i++)
> {
> t = new Thread(new ThreadStart(new zzz().abc));
> t.Start();
> t.Join();
>
> }
>
> }
> public void abc()
> {
> //Calls to DAL methods
>
> this.WriteToFile();
> }
> private void WriteToFile()
> {
> mut.WaitOne();
>
> using (StreamWriter sw = File.AppendText(@"d:\text.txt"))
>
> try
> {
>
> sw.WriteLine(AppDomain.GetCurrentThreadId());
```

Re: Multithreading – writing to same file C#

Re: Multithreading – writing to same file C#

```
> sw.Flush();
> sw.Close();
> }
> catch (Exception e)
> {
> Console.WriteLine("error:{0}", e);
> }
>
> finally
> {
> mut.ReleaseMutex();
> }
>
>
>
> }
> }
>
> The issue is that I have to use the Thread.Join() method else the main
> terminates. Hence not all threads execute. But if I use Thread.Join()
> then there are no concurrent users connecting. can anyone help me with
> this.
>
> Regards
> Mickey
>
```

• **Follow-Ups:**

◆ **Re: Multithreading – writing to same file C#**

◇ From: Mickey

• **References:**

◆ **Multithreading – writing to same file C#**

◇ From: mahendranepali

• Prev by Date: **Re: From ushot to byte[]**

• Next by Date: **Re: NEED HELP!!!**

• Previous by thread: **Re: Multithreading – writing to same file C#**

• Next by thread: **Re: Multithreading – writing to same file C#**

• Index(es):

◆ **Date**

◆ **Thread**