

Forum: [Hack Forums](#)

Thread: [Faster way loading lists](#)

Mr. Carter - December 13, 2014, 5:18 pm

Well decided to drop this, since this is a much faster way loading lists into list of strings rather than loading into listbox.

So, after public class form add
Code Dim list As New List(Of String)
To load items into list,
Code Using FD As New OpenFileDialog()
FD.Title = "Load a text file"
FD.Filter = "Text Files (*.TXT;)|*.TXT|All files (*.*)|*.*" If FD.ShowDialog =
Windows.Forms.DialogResult.OK Then
list.AddRange(IO.File.ReadAllLines(FD.FileName))
End If
End Using
To clear just simply do
Code list.clear
The other functions work like that too like (add, addrange, etc..).

Well not sure if I'm not missing anything tho, that's mostly for just loading and clearing the list.

Agent Raigar - December 13, 2014, 5:22 pm

StreamReader, reading line-by-line would be 100 times better than this.

SIM-1676753 - December 13, 2014, 5:54 pm

(Dec 13, 2014 06:22 PM)Agent Raigar Wrote: [\[CITE\]](#) StreamReader, reading line-by-line would be 100 ways better than this.

Spoiler [\(Click to View\)](#)

Spoiler [\(Click to View\)](#)

<http://referencesource.microsoft.com/#mscorlib/system/io/file.cs,5150c565446cd5fd>

The framework is essentially doing the same thing.

Cairns - December 13, 2014, 5:59 pm

Actually if the file is smaller it does read them quicker by using readalllines

However if you have a fairly large file the "readalllines" method just shits on itself and does not even attempt to load them.

Use a loop and read line by line using the streamreader. This way no matter how big the file is it will still be read

SIM-1676753 - December 13, 2014, 6:14 pm

(Dec 13, 2014 06:59 PM)Cairns Wrote: [\[CITE\]](#) Actually if the file is smaller it does read them quicker by using readalllines

However if you have a fairly large file the "readalllines" method just shits on itself and does not even attempt to load them.

Use a loop and read line by line using the streamreader. This way no matter how big the file is it will still be read

Microsoft uses a StreamReader in File.ReadAllLines as shown above.

NETExplorer - December 13, 2014, 7:28 pm

(Dec 13, 2014 07:14 PM)Mr. Trvp Wrote: [\[CITE\]](#) Microsoft uses a StreamReader in File.ReadAllLines as shown above.

If you use File.ReadAllLines: Lines -> List -> Array -> List

StreamReader: Lines -> List

Raz?r - December 13, 2014, 7:34 pm

You also might want to multithread the loop for greater speed.

You should get the best results with a multi-threaded streamreader.

Mr. Carter - December 14, 2014, 1:50 am

(Dec 13, 2014 08:34 PM)Raz?r Wrote: [\[CITE\]](#) You also might want to multithread the loop for greater speed.

You should get the best results with a multi-threaded streamreader.

it can go even faster lmao ? :D Well this is fast enough for me really, thanks to those C# open source crackers out there.

Cairns - December 14, 2014, 4:32 am

Good luck multi threading the reading of a stream reader. If you manage to do it, it will actually slow the process down. It will only make the reading of LARGE files responsive. But it will still take time to load them

idb. - December 14, 2014, 5:20 am

If it's not necessary to queue the entire list before beginning operation then you shouldn't. Spawn a new thread that will begin reading your text file until it reaches the end of it and will then signal that it has reached the end. While this is going on your can spawn your other worker threads to begin processing the information that has already been loaded. If no items are available in the queue, you check to see if the loading has been completed. If it has, you have reached the end of your list and can complete operation. If not you wait for more items to become available and continue with your operation. No real reason to be sitting around waiting for a gigantic list to load completely.

maruata - December 14, 2014, 5:34 am

Nice to read! Reading line by line is what I am agreed upon!