

Forum:[Hack Forums](#)

Thread: [Saving a list in textfile](#)

Tunabish - June 13, 2018, 10:44 am

I need to save a list in a textfile and load the lists back from the textfiles but I have no idea how I was told to use JSON but I don't think I'm allowed to as my uni requires it to be a textfile for some stupid reason.

Does anyone know how to do this? x_x

HFDone - June 13, 2018, 10:49 am

```
foreach (var item in listBox1.Items) {  
    File.AppendAllText("text.txt", item);  
}
```

```
string[] lines = File.ReadAllLines("text.txt");  
listBox1.Items.AddRange(lines);
```

Tunabish - June 13, 2018, 1:32 pm

(Jun 13, 2018 10:49 AM)HFDone Wrote: [\[CITE\]](#) foreach (var item in listBox1.Items) {
 File.AppendAllText("text.txt", item);
}

```
string[] lines = File.ReadAllLines("text.txt");  
listBox1.Items.AddRange(lines);
```

A list of objects which I need to load back on startup, not a listbox.

Nick (Name) - June 13, 2018, 2:24 pm

This is my stab at what I believe you're trying to accomplish. It's written in .NET 4.5 and uses async tasks.

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

Cr1p - June 14, 2018, 4:25 am

Bruh. If your objects are serializable into strings (which I assume they are because you think you can write them to a text file), use Newtonsoft.Json

```
var myList = new List<MyObject>();  
var myString = JsonConvert.SerializeObject(myList);  
File.WriteAllText(myString.....);
```

```
var x= File.ReadAllText(...);  
var myList = JsonConvert.DeserializeObject<List<MyObject>>(x);
```

Just realized you said not to use JSON.

In that case, there really isn't much we can provide for you unless you give us the object type you want to write to a file.

Moxxi - June 14, 2018, 7:22 am

assuming list of strings and console app, This should work for what you want

options are save, load, display, and exit

Spoiler ([Click to View](#))

0xDEAD10CC - June 14, 2018, 11:45 am

```
(Jun 13, 2018 10:49 AM)HFDone Wrote: \[CITE\] foreach (var item in listBox1.Items) {  
    File.AppendAllText("text.txt", item);  
}
```

```
string[] lines = File.ReadAllLines("text.txt");  
listBox1.Items.AddRange(lines);
```

Or, you could use WriteAllLines, since appending text without a line isn't going to work unless each listbox object to string contains it's own newline at the end, and the next time you write to the file it won't clear the first list.

OP:

And if it is a list of objects, why would you write them out line by line? Just serialize the information and deserialize it?

Tunabish - June 16, 2018, 2:55 am

Posting my solution here incase anyone else runs into the same problems.

I have a list of objects with people, which have a name, an address and an age.

In the class I have functions to get those.

```
Code    public void writePeople()  
    {  
        using (FileStream fileStream = new FileStream("people.txt", FileMode.OpenOrCreate,  
FileAccess.Write))  
        {  
            using (StreamWriter writer = new StreamWriter(fileStream))  
            {  
                foreach (var Person in People)  
                {  
                    writer.WriteLine(Person.getName + ":" + Person.getAddress + ":" + Person.getAge);  
                }  
                writer.Flush();  
            }  
        }  
    }  
}I write them all seperated by a ":" (you can use whatever you want)
```

Then I read them back by cutting off the string everytime it sees ":"

and assigning them to their variables so I can add them to the list again.

```
Code    public void readPeople()  
    {  
        People.Clear();  
  
        using (FileStream fileStream = new FileStream("people.txt", FileMode.OrCreate,  
FileAccess.Read))  
        {  
            using (StreamReader reader = new StreamReader(fileStream))
```

```

{
    string line;

    while ((line = reader.ReadLine()) != null)
    {
        string[] propArray = line.Split(':');

        string name = propArray[0];
        string address = propArray[1];
        int age = int.Parse(propArray[2]);

        // Add the person with these things.
        People.Add(
            new Person(name, age, address)
        );
    }
}

```

}you have to use FileMode.Create so it makes a new file, else it might keep the old list and not write a new one.

(atleast if you want to write a new list everytime).

Hope anyone is helped by this ?

Thanks for the people that tried to help me, and goodluck to those who need it!

CodeAssassin - June 16, 2018, 5:13 am

Would make sense to override the ToString() method and set your delimited string there

Example

```

Code public override string ToString() {

    return $"{Name}:{Address}:{Age}";
}

```

0xDEAD10CC - June 16, 2018, 12:00 pm

(Jun 16, 2018 05:13 AM)CodeAssassin Wrote: [\[CITE\]](#) Would make sense to override the ToString() method and set your delimited string there

Example

```

Code public override string ToString() {

    return $"{Name}:{Address}:{Age}";
}

```

Not really, because the ToString() method by convention isn't typically used for parsing, it's normally used for displaying an object in human readable form. I would separate it into another method to make it clear that the string output would be the format required for the parsing if this is the way he's choosing to go over serialization..