Store data from a C# application

Asked 16 years, 4 months ago Modified 12 years, 5 months ago

Viewed 10k times

16

I've recently taken up learning some C# and wrote a Yahtzee clone. My next step (now that the game logic is in place and functioning correctly) is to integrate some method of keeping stats across all the games played.

My question is this, how should I go about storing this information? My first thought would be to use a database and I have a feeling that's the answer I'll get... if that's the case, can you point me to a good resource for creating and accessing a database from a C# application?

Storing in an XML file actually makes more sense to me, but I thought if I suggested that I'd get torn apart ;). I'm used to building web applications and for those, text files are generally frowned upon.

So, going with an XML file, what classes should I be looking at that would allow for easy manipulation?

c# .net

Share

Improve this question

edited Jul 3, 2012 at 13:29

casperOne

Follow

74.5k ● 19 ● 188 ● 259

asked Aug 21, 2008 at 14:24

Justin Bennett

9,128 ● 2 ● 28 ● 30

9 Answers

Sorted by: Highest score (default) ⬍

17

Here is one idea: use Xml Serialization. Design your GameStats data structure and optionally use Xml attributes to influence the schema as you like. I like to use this method for small data sets because its quick and easy and all I need to do is design and manipulate the data structure.

```
using (FileStream fs = new FileStream(....))
{
    // Read in stats
    XmlSerializer xs = new XmlSerializer(typeof(GameSt
    GameStats stats = (GameStats)xs.Deserialize(fs);

    // Manipulate stats here ...

    // Write out game stats
    XmlSerializer xs = new XmlSerializer(typeof(GameSt
    xs.Serialize(fs, stats);

    fs.Close();
}
```

Share Improve this answer

Follow

answered Aug 21, 2008 at 14:42

Brian Ensink

11.2k ● 3 ● 51 ● 63

13

A database would probably be overkill for something like this - start with storing your information in an XML doc (or series of XML docs, if there's a lot of data). You get all that nifty XCopy deployment stuff, you can still use LINQ, and it would be a smooth transition to a database if you decided later you really needed performant relational query logic.

Share Improve this answer

Follow

answered Aug 21, 2008 at 14:31

Greg Hurlman

17.8k ● 6 ● 55 ● 87

7

A database may be overkill - have you thought about just storing the scores in a file?

If you decide to go with a database, you might consider SQLite, which you can distribute just like a file. There's an open source .NET provider - System.Data.SQLite - that includes everything you need to get started.

Accessing and reading from a database in .NET is quite easy - take a look at this question for sample code.

Share Improve this answer

Follow

edited May 23, 2017 at 12:17

Community Bot

1 ● 1

answered Aug 21, 2008 at 14:31

palmsey

5,832 ● 3 ● 38 ● 41

4

SQL Express from MS is a great free, lightweight version of their SQL Server database. You could try that if you go the DB route.

Alternatively, you could simply create datasets within the application and serialize them to xml, or you could use something like the newly minted Entity Framework that shipped with .NET 3.5 SP1

Share Improve this answer Follow

answered Aug 21, 2008 at 14:34

Jon Dewees
2,957 ● 6 ● 31 ● 38

3

I don't know if a database is necessarily what you want. That may be overkill for storing stats for a simple game like that. Databases are good; but you should not automatically use one in every situation (I'm assuming that this is a client application, not an online game).

Personally, for a game that exists only on the user's computer, I would just store the stats in a file (XML or binary - choice depends on whether you want it to be human-readable or not).

Share Improve this answer Follow

answered Aug 21, 2008 at 14:32

TheSmurf
15.6k ● 3 ● 41 ● 48

2

I'd recommend saving your data in simple POCOs and either serializing them to xml or a binary file, like Brian did above.

If you're hot for a database, I'd suggest Sql Server Compact Edition, or VistaDB. Both are hosted inproc within your application.

Share Improve this answer Follow

answered Aug 21, 2008 at 14:49

user1228

1

I would recommend just using a database. I would recommend using LINQ or an ORM tool to interact with the database. For learning LINQ, I would take a look at Scott Guthrie's posts. I think there are 9 of them all together. I linked part 1 below. If you want to go with an ORM tool, say nhibernate, then I would recommend checking out the Summer of nHibernate screencasts. They are a really good learning resource for nhibernate.

I disagree with using XML. With reporting stats on a lot of data, you can't beat using a relational database. Yeah, XML is lightweight, but there are a lot of choices for light weight relational databases also, besides going with a full blown service based implementation. (i.e. SQL Server Compact, SQLite, etc...)

- Scott Guthrie on LINQ
- Summer of nHibernate

Share Improve this answer Follow

answered Aug 21, 2008 at 14:47

Dale Ragan

18.3k ● 3 ● 55 ● 71

No idea why you've been down-voted so much. In this case, I wouldn't myself go to a database (even the lightweight ones you have described), but your point is valid and one day the original poster may have added enough to his stats to make a DB a more usable option (+1 to try and uncondemn this) – Ray Hayes Sep 25, 2008 at 13:47

1

You can either use the `System::Xml` namespace or the `System::Data` namespace. The first gives you raw XML, the latter gives you a handy wrapper to the XML.

Share Improve this answer

Follow

edited Jul 3, 2012 at 13:20

user142162

answered Aug 21, 2008 at 14:38

Skizz

71k ● 10 ● 74 ● 109

1

For this situation, the `[Serializable]` attribute on a nicely modelled `Stats` class and `XmlSerializer` are the way to go, IMO.

Share Improve this answer

Follow

edited Jul 3, 2012 at 13:21

user142162

answered Aug 21, 2008 at 15:17

Chris Marasti-Georg

34.6k ● 17 ● 93 ● 137