Object persistence strategy for desktop application

Asked 15 years, 11 months ago Modified 15 years, 5 months ago

Viewed 1k times

4

I am developing a Java based desktop application. There are some data generated from the application object model that I need to persist (preferably to a file). There is also a requirement to protect the persisted file so that others can't derive the object model details from the data. What's the best strategy for doing these? I was in the impression that these requirements are very common for desktop apps. However, I haven't been able to found much useful info on it. Any suggestion appreciated.

java desktop object-persistence

Share

Improve this question

Follow

asked Dec 26, 2008 at 16:31

hushan

6 Answers

Sorted by: Highest score (default)

Your question has two parts. 1st: How to persist data? 2nd: How to protect them?

2

There is a lot of ways how to persist data. From simple XML, java serialization to own data format. There is no way how to prevent revers engineering data just by "plain text". You can just make it harder, but not impossible. To make it quite impossible you need to use strong encryption and here comes a problem. How to encrypt data and don't reveal secure token. If you are distributing secure token with your application it is just a matter of time to find it and problem is solved. So entering a secure token during installation is not an option. If user has to authenticate to use application it should help, but it is the same problem. The next option is to use custom protected bijection algorithm to obfuscate data. And the last option is to do nothing just keep the data format private and don't publish them and obfuscate your application to prevent from reverse engineering.

At the best value comes simple obfuscation of data (XOR primenumber) with custom data format and obfuscated application.

Share Improve this answer

Follow

answered Dec 26, 2008 at 19:05

Rastislav Komara

1,721 ● 8 ● 17

2

If you don't need to modify this file you can serialize the object graph to a file. The contents are binary and they could only be read using the classes where they were written.

You can also use Java DB (shipped with java since 1.5 I think) and an ORM tool for that such as Hibernate.

EDIT

It is bundled since 1.6 http://developers.sun.com/javadb/

Share Improve this answer Follow

edited Dec 26, 2008 at 16:48

answered Dec 26, 2008 at 16:37

OscarRyz
199k ● 119 ● 396 ● 573

1 If you serialize the objects in a binary format, though, you do need to be careful to think first about the importance of versioning, compatibility, and future extensibility for your data... – reuben Dec 26, 2008 at 17:07

1

XStream works if you want to do simple xml reading and writing to a file. Xstream allows you to take any java object and write it to and read it from you file.

Share Improve this answer Follow

edited Jul 1, 2009 at 18:53

Scott Ivey
41.5k ● 21 ● 84 ● 118

answered Dec 26, 2008 at 16:54

Milhous
14.6k ● 16 ● 66 ● 83

Do I then encrypt the XML file for protection? It sounds like a viable solution besides plain old serialization. I will look more into it. Thanks. – hushan Dec 26, 2008 at 17:12

You could encrypt it, but as you are running code on the desktop, the user will always be able to decryt it(they are running your code). – Milhous Dec 26, 2008 at 19:20

0

I think "serialization" is the word:

http://java.sun.com/developer/technicalArticles/Programming/serialization/

Share Improve this answer

Follow

answered Dec 26, 2008 at 17:00

pistacchio
58.8k ● 110 ● 287 ● 431

Can serialized file be read without the class files (e.g. using byte code enhancement library)? – hushan Dec 26, 2008 at 17:09

This misses the requirement to hide the object model from scrutiny! – joel.neely Dec 26, 2008 at 18:22

0

If you really need the security implied in your statement ("...protect the persisted file so that others can't derive the object model details from the data."), I'd serialize the data in memory (to Java serialized form, XML, or whatever) and then encrypt that byte-stream to a file.

Share Improve this answer

Follow

answered Dec 26, 2008 at 18:23

joel.neely

30.9k • 9 • 57 • 64

0

You can try using an embedded database like Berkeley DB Java Edition (http://www.oracle.com/database/berkeley-db/je/index.html). Their direct persistent layer API will most likely suit your needs. The database contents are synced to files on disk. From just looking at the files directly, it's not easy to figure out the object model from the data. I've had good experiences with it, it's lightning fast and works well with desktop applications.

Share Improve this answer

Follow

answered Dec 26, 2008 at 19:23

mweiss

1,373 • 7 • 14

just a caution - the last I checked, this project is GPL (unlike the non-java edition of the Berkeley DB). It is a very nice DB - but if you aren't going fully open source, the licensing may be a problem – Kevin Day Dec 26, 2008 at 23:31