Subject: serialization
Posted by gprentice on Sun, 02 Apr 2006 11:18:26 GMT
View Forum Message <> Reply to Message

In the serialize reference example, this code appears to cleverly handle both the saving and restoring - is that correct?

```
 void Serialize(Stream& s)
 {
  s % q % text % data;
 }
```

Suppose you have 100 items to save/restore - is there any reason not to chain all together in one long expression?

Comparing with .ini file, suppose some objects get removed - is there any alternative to having "dummy" objects of the correct type so that you can still read in data that was saved with now obsolete items?

Graeme

Subject: Re: serialization
Posted by mirek on Sun, 02 Apr 2006 15:51:17 GMT
View Forum Message <> Reply to Message

gprentice wrote on Sun, 02 April 2006 07:18
In the serialize reference example, this code appears to cleverly handle both the saving and restoring - is that correct?

```
 void Serialize(Stream& s)
 {
  s % q % text % data;
 }
```

Yes.

Quote:
Suppose you have 100 items to save/restore - is there any reason not to chain all together in one long expression?

Usually not (see e.g. ide/idewin.cpp line 41

Quote:
Comparing with .ini file, suppose some objects get removed - is there any alternative to having "dummy" objects of the correct type so that you can still read in data that was saved with now obsolete items?

Well, not really. Binary serialization has its limitations. In fact, I would not recommend it for "robust" data storage, e.g. for storing documents. We are using it for e.g. configuration files, passing data over network or backuping data for implementation reasons (e.g. Cancel in dialogs can be implemented using serialization - before stating dialog, its content is serialized into String, Cancel serializes it back).

Mirek

Subject: Re: serialization
Posted by barpas on Wed, 05 Apr 2006 17:28:44 GMT
View Forum Message <> Reply to Message

halo everybody

i have problem with serialization - some time ago i writen application in upp where i use serialization mechanism to save options & parameters (there is a lot of serialize functions) - now i have problem user told that anybody can edit that files and read (change) information abaut program
so i need to code it - is there simple metod to do it fast (i have litle bit of time)

Subject: Re: serialization
Posted by mirek on Wed, 05 Apr 2006 18:50:11 GMT
View Forum Message <> Reply to Message

barpas wrote on Wed, 05 April 2006 13:28halo everybody

i have problem with serialization - some time ago i writen application in upp where i use serialization mechanism to save options & parameters (there is a lot of serialize functions) - now i have problem user told that anybody can edit that files and read (change) information abaut program
so i need to code it - is there simple metod to do it fast (i have litle bit of time)

Interesting...

Well, the most simple approach that comes to mind is to serialize using StringStream to String, then encrypt and save (and reverse for load).

Mirek

Subject: Re: serialization
Posted by nixnixnix on Thu, 20 Mar 2008 15:34:32 GMT
View Forum Message <> Reply to Message

Hi,

What would you recommend for robust data storage please?

At present I am thinking in terms of oversizing the storage for all my classes i.e. setting a size for each class for it to grow into.

Is there a way to search for specific "key" values in the stream which might then act as delimiters between classes?

I am getting close to launch and people are testing and already asking for backwards compatibility which I know from experience just leads to more and more complication.

If anyone has a solution, I would be extremely eager to hear it

Nick

Subject: Re: serialization
Posted by mirek on Thu, 20 Mar 2008 16:11:06 GMT
View Forum Message <> Reply to Message

nixnixnix wrote on Thu, 20 March 2008 11:34Hi,

What would you recommend for robust data storage please?

Depends. In theide, some text format with C lexical basics is prefered (e.g. for .upp files).

Maybe XML.

Quote:
Is there a way to search for specific "key" values in the stream which might then act as delimiters between classes?

No.

Quote:

I am getting close to launch and people are testing and already asking for backwards compatibility which I know from experience just leads to more and more complication.

Hehe, I know that pretty well

Mirek

Subject: Re: serialization
Posted by zsolt on Thu, 20 Mar 2008 17:42:10 GMT
View Forum Message <> Reply to Message

Quote:If anyone has a solution, I would be extremely eager to hear it
I'm using per class versioning in serialize methods and I have no problems with it at all.
I have to increase version number and put some "if"s into my code after changing classes.

Subject: Re: serialization
Posted by nixnixnix on Thu, 20 Mar 2008 20:54:04 GMT
View Forum Message <> Reply to Message

Ok thanks. This is what I was using before but if thats the standard method then its all good.

XML is not really suitable because even in binary my saved documents can be easily 100s of MB and more. Also lots of my classes contain Arrays so they don't have fixed sizes.

Ach well I just need to "suck it up" I guess

Cheers,

Nick

Subject: Re: serialization
Posted by chickenk on Thu, 20 Mar 2008 21:23:59 GMT
View Forum Message <> Reply to Message

What about YAML? (http://www.yaml.org)

It can embed base64 encoded binary chunks...

I believe a C implementation is present here: http://pyyaml.org/wiki/LibYAML

I'm quite fond of YAML when talking about serialization tasks, but maybe it's not suited for you... If I had more time, I would consider making a C++ wrapper around libYAML using U++ technology.

If anyone interested...

Lionel

Subject: Re: serialization
Posted by tojocky on Sat, 12 Jul 2008 09:43:19 GMT
View Forum Message <> Reply to Message

chickenk wrote on Thu, 20 March 2008 23:23What about YAML? (http://www.yaml.org)

It can embed base64 encoded binary chunks...

I believe a C implementation is present here: http://pyyaml.org/wiki/LibYAML

I'm quite fond of YAML when talking about serialization tasks, but maybe it's not suited for you... If I had more time, I would consider making a C++ wrapper around libYAML using U++ technology.

If anyone interested...

Lionel

Good idea!

Subject: Re: serialization
Posted by captainc on Mon, 14 Jul 2008 14:26:12 GMT
View Forum Message <> Reply to Message

chickenk wrote on Thu, 20 March 2008 17:23What about YAML? (http://www.yaml.org)

It can embed base64 encoded binary chunks...

I believe a C implementation is present here: http://pyyaml.org/wiki/LibYAML

I'm quite fond of YAML when talking about serialization tasks, but maybe it's not suited for you... If I had more time, I would consider making a C++ wrapper around libYAML using U++ technology.

If anyone interested...

Lionel
Yes, definitely interested!

Subject: Re: serialization
Posted by koldo on Mon, 18 May 2020 18:09:31 GMT

Does anybody has used a YAML reader/writer?