Files and Streams

4-22-2004

Opening Discussion

- What did we talk about last class?
- Do you have any questions about the assignment?
- Writing and reading one byte at a time gives you enough power to do anything, but it isn't optimal for most things. We'd like to be able to write other things as well. How would you go about doing that?

Binary Files

- Most of the time, the way that we want to store real data in files is in binary format. For everything but strings, this takes a lot less space than storing string equivalents and is faster to read and write.
- With a binary file, we can write ints, doubles, and other primitives as well as strings. The files won't be human editable, but we can write code to read them back in.

Making Streams from Streams

- One of the keys to being able to use the java.io library is to notice that many stream types have constructors that you pass other streams to.
- These create new streams that have different functionality and use the stream that is passed to them to send the data. In effect, you are wrapping one stream inside another to get different functionality for the same source/dest.

Data I/O Streams

- To do basic binary I/O in Java we use the DataInputStream and DataOutputStream classes. These can't exist "on their own". We use them to wrap another stream that actually goes somewhere.
- These classes provide us with the functionality to read and write basic types.
- Let's look at these classes real quick.

Object Streams

- We can write pretty much any class out to a stream by writing one component at a time, but doing so can be painful. Sometimes we want to be able to write an object as a single entity.
- In Java we can do this with ObjectInputStream and ObjectOutputStream.
- This is something that most languages don’t support.

Serialization

- Writing objects to streams is also called serializing them. The object streams can only work with two types of data: primitives and Serializable. For an object to be serialized it must be Serializable and all its members must be either primitives or Serializable.
- Members that are declared transient are not serialized.

More on Serialization

- Serialization is an incredibly powerful tool. When combined with reflection in Java it lets us do things that aren't possible in most languages.
- "With great power comes great responsibility." This is true in comics and in programming. You have seen some of the difficulties of using serialization and there are many more.

Code

- Let’s write some code that will write an object in binary format, then do the same thing with serialization.

Minute Essay

- What are the benefits of “wrapping” streams around one another? Another stream type that does this is the buffered stream. How can we use this to get better performance for a data stream?