Java IO
and
C++ Streams

October 22, 2004

Outline

- Java IO
  - InputStream/OutputStream
  - FilterInputStream/FilterOutputStream
  - DataInputStream/DataOutputStream
  - Reader/Writer
  - Encodings
- C++ Streams

java.io - Intro

Two main abstractions:

- streams of bytes
  - InputStream
    - abstract class
    - is the superclass of all classes representing an input stream of bytes
  - OutputStream
    - abstract class
    - is the superclass of all classes representing an output stream of bytes
- streams of characters

java.io - Intro

Two main abstractions:

- streams of bytes
- streams of characters
  - Reader
    - abstract class
    - is the superclass of all classes representing an input stream of characters
  - Writer
    - abstract class
    - is the superclass of all classes representing an output stream of characters

OutputStream

- must always provide at least a method that writes one byte of output
- methods:

```
void close()
void flush()
void write(byte[] b)
void write(byte[] b, int off, int len)
abstract void write(int b)
```

- subclasses

OutputStream (cont)

- subclasses:
  - ByteArrayOutputStream
    - data is written into a byte array
    - the buffer automatically grows as data is written to it
  - FileOutputStream
    - writing data to a File or to a FileDescriptor
    - meant for writing streams of raw byte
  - FilterOutputStream
  - ObjectOutputStream
  - PipedOutputStream

OutputStream (cont)

- subclasses:
  - ByteArrayOutputStream
  - FileOutputStream
  - FilterOutputStream
    - superclass of all classes that filter output streams
    - subclasses: *BufferedOutputStream*, CheckedOutputStream, CipherOutputStream, *DataOutputStream*, DeflaterOutputStream, DigestOutputStream, PrintStream
  - ObjectOutputStream
  - PipedOutputStream

OutputStream (cont)

- subclasses:
  - ByteArrayOutputStream
  - FileOutputStream
  - FilterOutputStream
  - ObjectOutputStream
    - writes primitive data types and graphs of Java objects
    - only objects that support the java.io.Serializable interface can be written to streams
    - primitive data types can also be written
  - PipedOutputStream
    - can be connected to a piped input stream to create a communications pipe

FilterOutputStream

- writes data to another OutputStream
- two kind of subclasses:
  - processing of the data:
    - BufferedOutputStream
    - CheckedOutputStream
    - CipherOutputStream
    - DeflaterOutputStream
      - GZIPOutputStream
      - ZipOutputStream
    - DigestOutputStream
  - provides an expanded interface:
    - DataOutputStream
    - PrintStream: polymorphic print and println, printf

DataOutputStream

Extended interface provided by DataOutputStream:

```
void  writeBoolean(boolean v)
void  writeByte(int v)
void  writeBytes(String s)
void  writeChar(int v)
void  writeChars(String s)
void  writeDouble(double v)
void  writeFloat(float v)
void  writeInt(int v)
void  writeLong(long v)
void  writeShort(int v)
void  writeUTF(String str)
```

OutputStream - Example

```
FileOutputStream fileOut =
            new FileOutputStream("myfile");
// add buffering
BufferedOutputStream bufOut =
            new BufferedOutputStream(fileOut);
// add compression
GZIPOutputStream gzOut =
            new GZIPOutputStream(bufOut);
// interface for binary data
DataOutputStream out =
            new DataOutputStream(gzOut);
// some writing
out.writeInt(n);
out.writeDouble(d);
```

OutputStream - What if?

```
FileOutputStream fileOut =
            new FileOutputStream("myfile");
BufferedOutputStream bufOut =
            new BufferedOutputStream(fileOut);
GZIPOutputStream gzOut =
            new GZIPOutputStream(bufOut);
DataOutputStream out1 =
            new DataOutputStream(gzOut);
DataOutputStream out2 =
            new DataOutputStream(bufOut);

out1.writeInt(n);
out2.writeDouble(d);
```

OutputStream - What if?

```
FileOutputStream fileOut =
                new FileOutputStream("myfile");
BufferedOutputStream bufOut =
                new BufferedOutputStream(fileOut);
GZIPOutputStream gzOut =
                new GZIPOutputStream(bufOut);
DataOutputStream out1 =
                new DataOutputStream(gzOut);
DataOutputStream out2 =
                new DataOutputStream(bufOut);

out1.writeInt(n);
out2.writeDouble(d);
```

The output will be corrupted!

OutputStream - A common construction

```
DataOutputStream out =
  new DataOutputStream(
    new GZIPOutputStream(
      new BufferedOutputStream(
        FileOutputStream("myfile"))));

out.writeInt(n);
out.writeDouble(d);
```

InputStream

- similar with OutputStream but in reverse
- subclasses:
  - ByteArrayInputStream
  - FileInputStream
  - FilterInputStream
  - ObjectInputStream
  - PipedInputStream
  - SequenceInputStream
    - represents the logical concatenation of other input streams

The other world: characters

- Writer
  - methods that a subclass must implement are: write(char[], int, int), flush(), and close()
  - subclasses: BufferedWriter, CharArrayWriter, FilterWriter, OutputStreamWriter, FileWriter, PipedWriter, PrintWriter, StringWriter
- Reader
  - methods that a subclass must implement are read(char[], int, int) and close()
  - subclasses: BufferedReader, CharArrayReader, FilterReader, InputStreamReader, PipedReader, StringReader

What is the big difference?

The other world: characters

- Writer
  - methods that a subclass must implement are: write(char[], int, int), flush(), and close()
  - subclasses: BufferedWriter, CharArrayWriter, FilterWriter, **OutputStreamWriter**, FileWriter, PipedWriter, PrintWriter, StringWriter
- Reader
  - methods that a subclass must implement are read(char[], int, int) and close()
  - subclasses: BufferedReader, CharArrayReader, FilterReader, **InputStreamReader**, PipedReader, StringReader

What is the big difference? Encodings!

InputStreamReader/OutputStreamWriter

- InputStreamReader
  - is a bridge from byte streams to character streams
  - reads bytes and decodes them into characters using a specified charset
- OutputStreamWriter
  - the opposite of InputStreamReader

Encodings

- JDKTM 5.0 Documentation
  → Tool Docs
    → Internationalization Tools
      → native2ascii
        → Supported Encodings
- docs/guide/intl/encoding.doc.html

Another Universe:
C++ Streams

Reference

The C++ Programming Language, Third Edition by Bjarne Stroustrup. Published by Addison Wesley Longman, Inc.

Standard Streams

Several standard streams are declared in <iostream>:

```
ostream cout      // standard output stream
                  //                    of char
ostream cerr      // standard unbuffered output
                  //  stream for error messages
ostream clog      // standard output stream for
                  //                error messages
wostream wcout;   // wide stream ver. of cout
wostream wcerr;   // wide stream ver. of cerr
wostream wclog;   // wide stream ver. of clog
```

istream

Provides:

- »
- get
- getline
- read
- gcount
- putback
- tie

ostream

Provides:

- «
- put
- write

State

```
bool good() const;    // next op. might succeed
bool eof()  const;    // end of input seen
bool fail() const;    // next op. will fail
bool bad()  const;    // stream is corrupted
iostate rdstate() const; // get io state flags
void clear(iostate f = goodbit);
                      // set io state flags
void setstate(iostate f) {
  clear(rdstate()|f); // add f to iostate flags
}
operator void*() const; // nonzero if !fail()
bool operator!() const {
  return fail();
}
```

Formating

```
static const fmtflags
skipws,        // skip whitespace on input
left           // pad after value
right,         // pad before value
internal,      // pad between sign and value
boolalpha,     // use symbolic representation
dec,           // base 10 output (decimal)
hex,           // base 16 output (hexadecimal)
oct,           // base 8 output (octal)
scientific,    // d.dddddddEdd
fixed,         // dddd.dd
showbase,      // prefix oct by 0 and hex by 0x
showpoint,     // print trailing zeros
showpos,       // explicit '+' for positive ints
uppercase;     // 'E', 'X' rather than 'e', 'x'
```

Formating (cont)

```
// read flags
fmtflags flags() const;
// set flags
fmtflags flags(fmtflags f);
// add flags
fmtflags setf(fmtflags f) {
  return flags(flags()|f);
}
// clear flags
void unsetf(fmtflags mask) {
  flags(flags()&~mask);
}
```

Manipulators

- declared in <iomanip>, namespase std
- Manipulators:
  - boolalpha, noboolalpha
  - showbase, noshowbase
  - showpoint, noshowpoint
  - showpos, noshowpos
  - skipws, noskipws, ws
  - uppercase, nouppercase
  - internal
  - left, right
  - dec, hex, oct
  - fixed, scientific

Manipulators (cont)

- Manipulators (cont):
  - endl, ends
  - flush
  - resetioflags, setioflags
  - setbase
  - setfill
  - setprecision
  - setw

Other kinds of streams

- File Streams
  - <fstream>
  - ifstream, ofstream, fstream
- String Streams
  - <sstream>
  - istringstream, ostringstream, stringstream
  - wistringstream, wostringstream, wstringstream
- Stream Buffers

That’s all!

Have a nice weekend!