一、IO历史背景

Linux是第一个将设备抽象为文件的操作系统，在Linux中所有的外部设备都可以用读取文件的方法读取，这样编程人员就可以以操作文件的方法操作任何设备。C++在IO方面也做了一些改进――引进了流的概念，我们可以通过cin、cout读写一些对象。Java语言在IO设计方面取得较大的成功，它是完全面向对象的，主要采用装饰器模式避免大量的类，包括了最大的可能性，提供了较好的扩展机制……

二、Java IO的简单分类

1. IO 和 NIO

Java IO一般包含两个部分：1.java.io包中阻塞式IO；2.java.nio包中的非阻塞式IO，通常称为New IO。

(1) 阻塞式IO

传统JavaIO是基于阻塞I/O模型。这意味着，当一个线程调用read() 或 write()时，该线程被阻塞，直到有一些数据被读取，或数据完全写入。该线程在此期间不能再干任何事情了。

(2) 非阻塞式IO

① Java New IO的非堵塞技术主要采用了Observer模式，就是有一个具体的观察者和＝监测IO端口，如果有数据进入就会立即通知相应的应用程序。这样我们就避免建立多个线程，同时也避免了read等待的时间。

② Java NIO的非阻塞模式，使一个线程从某通道发送请求读取数据，但是它仅能得到目前可用的数据，如果目前没有数据可用时，就什么都不会获取。而不是保持线程阻塞，所以直至数据变的可以读取之前，该线程可以继续做其他的事情。 非阻塞写也是如此。一个线程请求写入一些数据到某通道，但不需要等待它完全写入，这个线程同时可以去做别的事情。 线程通常将非阻塞IO的空闲时间用于在其它通道上执行IO操作，所以一个单独的线程现在可以管理多个输入和输出通道（channel）。

2. 阻塞式IO基本分类

2.1 Java的IO主要包含三个部分：

(1) 流式部分――IO的主体部分；

(2) 非流式部分――主要包含一些辅助流式部分的类，如：File类、RandomAccessFile随机存储文件类和FileDescriptor等类；

(3) 文件读取部分的与安全相关的类，如：SerializablePermission类。以及与本地操作系统相关的文件系统的类，如：FileSystem类和Win32FileSystem类和WinNTFileSystem类。

2.2 流式部分

(1) 流式部分可以概括为：两个对应一个桥梁。

① 两个对应指：1.字节流（Byte Stream）和字符流（Char Stream）的对应；2.输入和输出的对应。

② 一个桥梁指：字节流和字符流之间的转换。对应于输入和输出为InputStreamReader和OutputStreamWriter。

③ 在流的具体类中又可以具体分为：1.介质流（Media Stream或者称为原始流Raw Stream）――主要指一些基本的流，他们主要是从具体的介质上，如：文件、内存缓冲区（Byte数组、Char数组、StringBuffer对象）等，读取数据；2.过滤流（Filter Stream）――主要指所有FilterInputStream/FilterOutputStream和FilterReader/FilterWriter的子类，主要是对其包装的类进行某些特定的处理，如：缓存等。

三、IO中的流式部分

1. 流的特点

流具有最基本的特点：“One dimension , one direction .” 即流是一维的，同时流是单向的。

2. 非流式特殊情况

(1) 背景

可能有些场合我们需要在文件中随机插入数据、在流中来来回回地执行某些操作，这时候我们绝对不可以使用流相关的对象。

(2) 解决方法：

一个单独的类RandomAccessFile，它可以完成打开、关闭文件、以基本数据类型的方式读取数据、读取下一个行、以UTF等格式读取数据、写入各种类型的数据、比较特殊的是他可以通过文件指针的seek方法让文件指针移到某个位置，可以通过getFilePointer方法得到当前指针的位置、可以通过length（）方法得到当前文件的容量、通过getFD得到FileDescriptor对象，通过getChannel方法得到FileChannel对象，从而和New IO整合。

3. IO中的输入字节流

3.1 所有相关类

(1) InputStream //所有输入字节流的父类（抽象类）

ByteArrayInputStream // Byte数组

FileInputStream // StringBuffer

StringBufferInputStream //本地文件

FilterInputStream //装饰流

BufferedInputStream

DataInputStream

LineNumberInputStream

PushbackInputStream

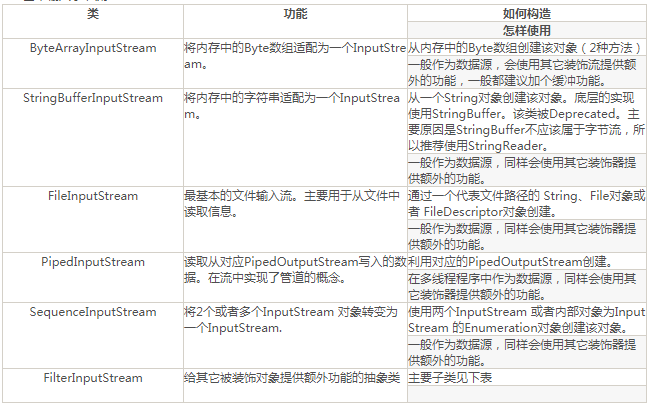
ObjectInputStream //装饰流

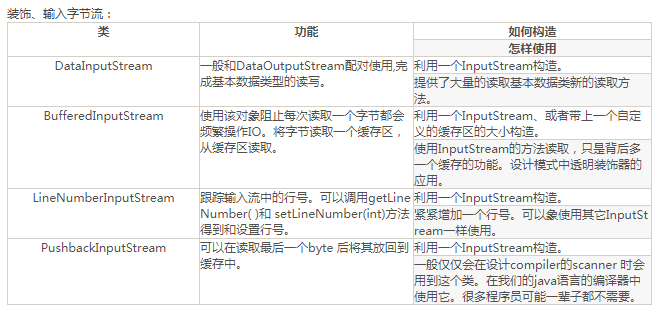
PipedInputStream //从与其它线程共用的管道中读取数据

SequenceInputStream

① ByteArrayInputStream、StringBufferInputStream、FileInputStream是三种基本的介质流，它们分别将Byte数组、StringBuffer、和本地文件中读取数据。

② ObjectInputStream和所有FilterInputStream的子类都是装饰流（装饰器模式的主角）。





4. IO中的输出字节流

(1) OutputStream

ByteArrayOutputStream // Byte数组

FileOutputStream //本地文件

FilterOutputStream

BufferedOutputStream

DataOutputStream

PrintStream

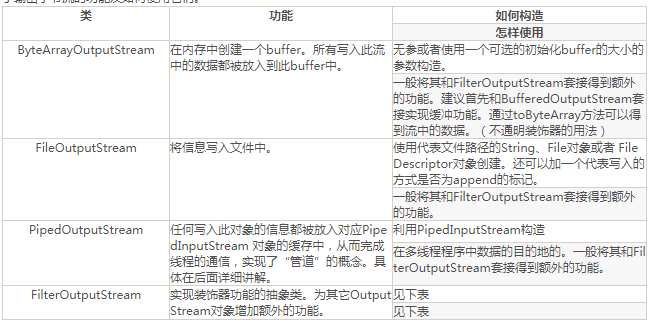
ObjectOutputStream

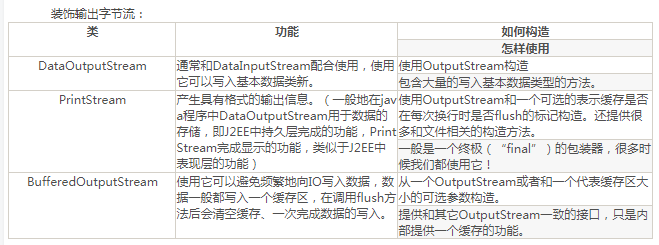
PipedOutputStream

① OutputStream是所有的输出字节流的父类，它是一个抽象类。

② ByteArrayOutputStream、FileOutputStream是两种基本的介质流，它们分别向Byte数组、 和本地文件中写入数据。PipedOutputStream是向与其它线程共用的管道中写入数据，

③ ObjectOutputStream和所有FilterOutputStream的子类都是装饰流。





5. IO中的输入字符流

(1) Reader

BufferedReader //一个装饰器，它和其子类负责装饰其它Reader对象

LineNumberReader

PipedReader //从与其它线程共用的管道中读取数据

StringReader // String

CharArrayReader // Char数组

FilterReader //所有自定义具体装饰流的父类

PushbackReader

InputStreamReader //连接字节流和字符流的桥梁，它将字节流转变为字符流

FileReader //常用的工具类将，FileInputStream转变为Reader

① CharReader、StringReader是两种基本的介质流，它们分别将Char数组、String中读取数据。

② Reader是所有的输入字符流的父类，它是一个抽象类。

6. IO中的输出字符流

(1) Writer

BufferedWriter

CharArrayWriter // Char数组

StringWriter // String

FilterWriter

OutputStreamWriter //OutputStreamWriter是OutputStream到Writer转换的桥梁

FileWriter

PipedWriter //向与其它线程共用的管道中写入数据

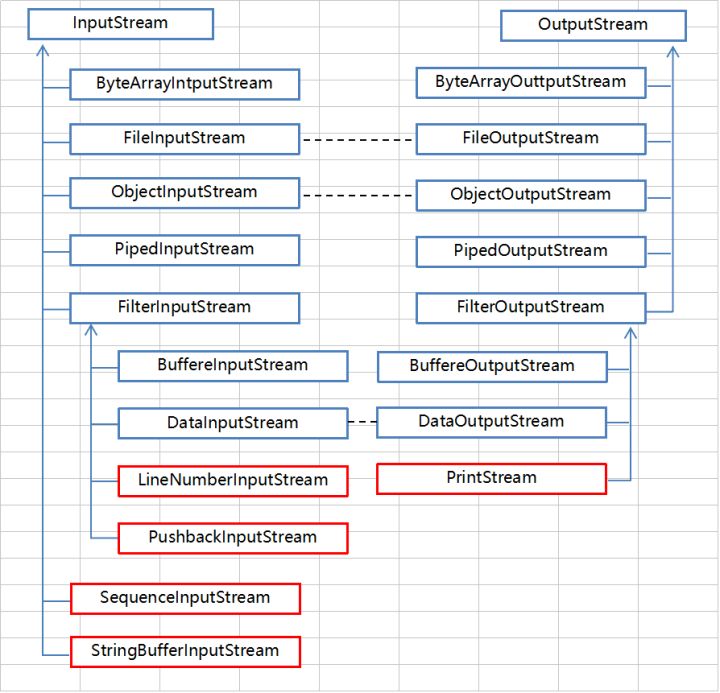
PrintWriter //格式化写入

① Writer是所有的输出字符流的父类，它是一个抽象类。

② CharArrayWriter、StringWriter是两种基本的介质流，它们分别向Char数组、String中写入数据。

7. 总结

(1) 字节流



注意： 图中蓝色为主要对应部分，红色为不对应部分，黑色的虚线部分代表这些流一般需

要搭配使用

(2) 字符流

