PrintWriter和BufferedWriter的区别

BufferedWriter:将文本写入字符输出流，缓冲各个字符从而提供单个字符，数组和字符串的高效写入。通过write()方法可以将获取到的字符输出，然后通过newLine()进行换行操作。BufferedWriter中的字符流必须通过调用flush方法才能将其刷出去。并且BufferedWriter只能对字符流进行操作。如果要对字节流操作，则使用BufferedInputStream。

PrintWriter:向文本输出流打印对象的格式化表示形式(Prints formatted representations of objects to a text-output stream)。PrintWriter相对于BufferedWriter的好处在于，如果PrintWriter开启了自动刷新，那么当PrintWriter调用println，prinlf或format方法时，输出流中的数据就会自动刷新出去。PrintWriter不但能接收字符流，也能接收字节流。

Socket编程中,尽量用PrintWriter取代BufferedWriter，下面是PrintWriter的优点：

1. PrintWriter的print、println方法可以接受任意类型的参数，而BufferedWriter的write方法只能接受字符、字符数组和字符串；

2. PrintWriter的println方法自动添加换行，BufferedWriter需要显示调用newLine方法；

3. PrintWriter的方法不会抛异常，若关心异常，需要调用checkError方法看是否有异常发生；

4. PrintWriter构造方法可指定参数，实现自动刷新缓存（autoflush）；

5. PrintWriter的构造方法更广。

-------------------------------------------------------------------

总结：

在使用BufferedReader中的readLine方法接收BufferedWriter中的字符流时，由于readLine是在读取到换行符的时候才将整行字符返回，所以BufferedWriter方法在录入一段字符后要使用newLine方法进行一次换行操作，然后再把字符流刷出去。而PrintWriter由于可以开启自动刷新，并且其中的println方法自带换行操作。所以代码实现起来要比BufferedWriter简单一些。

PrintWriter和BufferedWriter都是继承java.io.Writer，所以很多功能都一样。不过PrintWriter提供println()方法可以写不同平台的换行符，而BufferedWriter可以任意设定缓冲大小。OutputStream可以直接传给PrintWriter（BufferedWriter不能接收）,如：

PrintWriter out = new PrintWriter(new BufferedOutputStream(

new FileOutputStream("foo.out")));

或者用OutputStreamWriter来将OutputStream转化为Wrtier．这时就可以用BufferedWriter了。JDK API documents中都写的很清楚了。

PrintStream 和 PrintWriter 的区别

注意：当PrintWriter 调用println 时必须加上flush 才能进行通讯

区别1：

rintStream类和PrintWriter类

PrintStream主要操作byte流，而PrintWriter用来操作字符流。读取文本文件时一般用后者。

java的一个字符（char）是16bit的，一个BYTE是8bit的

PrintStrean是写入一串8bit的数据的。

PrintWriter是写入一串16bit的数据的。

String缺省是用UNICODE编码，是16bit的。因此用PrintWriter写入的字符串，跨平台性好一些吧。

PrintStream的可能会出现字符集乱码吧。

PrintStream是用来操作byte,

PrintWriter是用来操作Unicode,

一般需要处理中文时用PrintWriter好了

Stream用于二进制文件（非文本）

Writer/Reader用于文本文件（虽然也是二进制，不过是按照一定的字符编码规则，不像前者）

当然Stream也可用于文本，只不过比writer/reader来的麻烦

PrintStream主要操作byte流，而PrintWriter用来操作字符流。读取文本文件时一般用后者。

区别2：

java 输出流中的PrintStream 和 PrintWriter有什么区别，好像很多场合都能混用

比如用于网络间通信，发送消息时，可用PrintStream 也可用 PrintWriter

eg:

Socket client;

PrintStream output = new PrintStream(client.getOutputStream());

output.println("aaa");

output.flush();

把上例中的 PrintStream 换成 PrintWriter效果是一样的。Java I/O流分为两类，字节流和字符流。字节流是指InputStream/OutputStream及其子类，字符流是指Reader/Writer及其子类。这两类I/O流的class hierarchy基本上是对等的，InputStreamReader/OutputStreamWriter是InputStream/OutputStream和Reader/Writer之间的桥梁。

PrintStream是OutputStream的子类，PrintWriter是Writer的子类，两者处于对等的位置上，所以它们的API是非常相似的。PrintWriter实现了Writer的所有print方法。对于PrintStream，所有println方法都在内部调用相应的print方法，比如println(char x)的方法体就是调用print(x);再写入一个平台相关的换行符。PrintStream构建时会在内部new一个BufferedWriter，所有print方法都在内部调用这个Writer的write方法（write(String)或write(char[])）——对于print(char[])，直接调用write(char[])；对于其他的print方法，先用String.valueOf获得参数的字符串表示，然后调用write(String)。

对于PrintWriter，所有println方法也都在内部调用print方法,然后再调用一个没有参数的println()来换行，print方法调用write方法。传入OutputStream时，PrintWriter会在内部构造一个BufferedWriter；而传入Writer时，PrintStream内部直接使用该Writer，此时PrintWriter是否进行了缓冲全看该Writer。

所以，对于使用print系列方法的场合，二者没什么区别。

但是，PrintStream是字节流，它有处理raw byte的方法，write(int)和write(byte[],int,int)；PrintWriter是字符流，它没有处理raw byte的方法。PrintStream和PrintWriter的auto flushing机制有点不同，前者在输出byte数组、调用println方法、输出换行符或者byte值10（即\n）时自动调用flush方法，后者仅在调用println方法时发生auto flushing。

区别3：

PrintStream类是过滤器类中一个不可忽视的成员，最基本的标准输出就要借助于它——我们常用的System.out变量就是PrintStream实例。与之对应的字符流类是PrintWriter类。

　　PrintStream有两个构造函数(在新版ＡＰＩ中已标记为过时)：

　　■public PrintStream(OutputStream out)

　　■public PrintStream(OutputStream out,boolean autoFlush)

其中，autoFlush置为true时，每当输出遇到换行符，缓冲区的内容就被强制全部输出，如同调用了一次flush()。但要注意，如果没遇到换行符，还是会有数据“憋”在缓冲区里。

　　方法(已熟悉的就不解释)：

　　■public void write(int b)

　　■public void write(byte b,int offset,int length)

　　■public void flush()

　　■public void close()

　　■public void print(Object obj)

　　这个方法功能是 非常强大的，它可以输出任何对象，而不必另加说明。此外print()方法有许多重载形式，即有多种参数。它们是字符串(String)、字符数组 (char[])、字符(char)、整数(int)、长整数(long)、浮点数(float)、双精度浮点数(double)、布尔值 (boolean)。其中，输出多个数单位的print()方法(也就是指参数为String和char[]的)是同步(synchronized)方 法。

　　■public void println()输出一个换行符。

　　■public synchronized void println(Object obj)

　　println()方法有9个重载形式，几乎就是print()方法的翻版。唯一的区别在于println()方法都是同步的。

　　■public boolean checkError()

　　检查输出过程中有什么错误，如有，返回true值。只要输出流中出现一次错误，则出错后的任意对checkError()的调用均会返回真值。

　　下面介绍PrintWriter类。

　　如同第二节中所说，PrintWriter是JDK1.1版增加了与字节流I/O相对应的字符流I/O。但是，为了保持兼容性，原先的类几乎没有改 动。再加之调试的需要，PrintStream类被保留，并且System类中的成员变量out、err仍作为它的对象。然而，PrintWriter用 于大多数输出比PrintStream更为合适。因此1.1版的API中建议新开发的代码使用PrintWriter类，并将 PrintStream类的两个构造函数标记为过时。这样，虽然使用System.out输出不会产生问题，在程序中创建新的PrintStream对象 时却会产生编译时的警告。

　　PrintWriter类与PrintStream类的方法是对应的。有一个不同之外需提请读者注意，就是当前者的自动清空缓冲区的功能被使能时(构 造函数中autoFlush置为true)，仅当println()方法被调用时才自动清缓冲区，而不是像PrintStream一样遇到一个换行符就清 缓冲。 PrintStream，它可以将Java的基本资料型态等资料，直接转换为系统预设编码下对应的字元，再输出至OutputStream中，而这边要介 绍的 PrintWriter其功能上与PrintStream类似，除了接受OutputStream之外，它还可以接受Writer物件作为输出的对象，当 您原先是使用Writer物件在作处理 ，而现在想要套用println()之类的方法时，使用PrintWriter会是比较方便的作法。