java字符串无结束标记

文件流结束标记为-1

字符流：

FileReader

FIileWriter

BufferedReader

BufferedWriter

字节流：

FIileInputStream

FIileOutputStream

BufferedInputStream

BufferedOutputStream

和文件相关的流对象 一般都是比较重要的流对象

转换流：可以改变编码方式 默认采用GBK

读写到正确文件数据需要统一编码方式

不同编码方式 同样数据 文件大小可能不一样

GBK包含全部中文字符，  
UTF-8则包含全世界所有国家需要用到的字符。

InputStreamReader: 字节流转化为字符流

OutputStreamWriter:

System.in：对应标准输入设备，键盘

System.out：对应标准输出设备，控制台

System.setIn() 改变默认输入输出流

System.setOut()

总结： 如何选择使用流对象 举例：将一个mp3文件复制生成另一个文件

1. 确定源和目的（确定使用什么体系）

源：输入流 InputStream

目的：输出流 OutputStream

是否使用缓冲区：是 BufferedInputStream BufferedOutputStream

1. 确定是否为纯文本（数据类型）

字节流mp3文件 使用输入流FileInputStream

mp3文件 使用输出流 FileOutputStream

1. 确定输入输出设备

字节输入输出流转化为字符输入输出流： InputStreamReader ：

就可以使用字符流的readLine方法

缓冲区流类 方法：readLine();（

使用流之后 记得关闭 释放资源

1.使用字节流在没有使用缓冲区的情况下 输出不需要刷新

2.如何实现大文件的复制

3.



4.

A B均使用字节流输出缓冲区类（BufferedOutputStream）\*

<a> byte buf[]=new byte[1024];

while((len=bi.read(buf))!=-1)

{

bo.write(buf,0,len);

}

<b> while((byt=bi.read()!=-1))

{

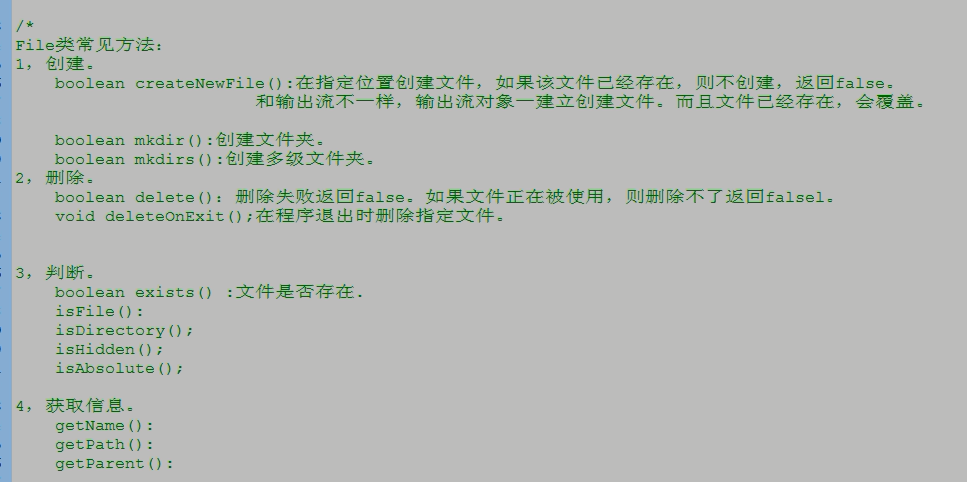
bo.write(byt);

}

代码a的执行效率比代码b的执行效率更高

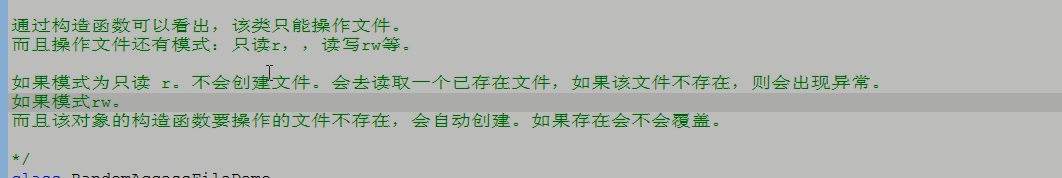
5.自我实现字节流缓冲区19天14课

List Hashtable map集合 回去看一看





RandomAccessFile：随机读写文件



printWriter和PrintStream：可以直接操作输入流和文件

序列流 SEquenceInputStream： 对多个流进行合并

ObjectInputStream与ObjectOutputStream：被操作对象需要实现serializable

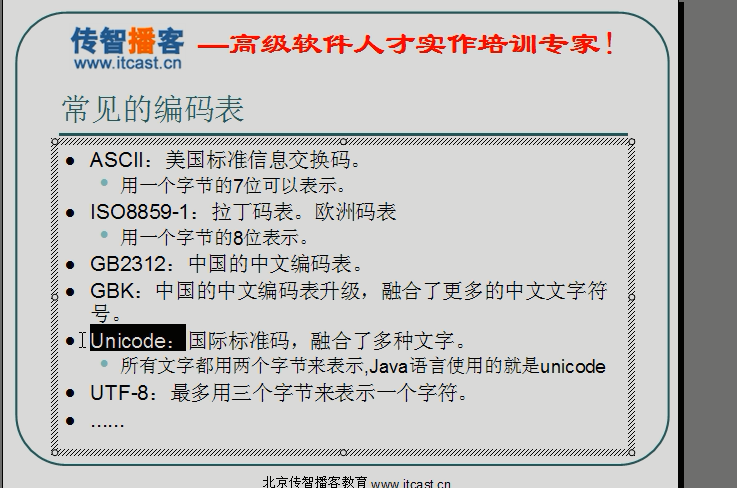
DataInputStream： 读取基本数据类型

ByteArrayInputStream和ByteArrayInputStream：

关闭该流无效，因为不涉及底层资源调用，也不会产生IOException

目的流的缓冲区可自动增长

编码：开发中经常遇到编码问题



Java char使用的事unicode编码方式