ctrl+F 查找

2-1 File类常用API介绍

java.io.File类用于表示文件(目录)

File类只用于表示文件(目录)的信息(名称、大小等)，不能用于文件内容的访问

3-1RansdomAccessFile基本操作

RandomAccessFile java提供的对文件内容的访问,既可以读文件,也可以写文件.

RandomAccessFile支持随机访问文件,可以访问文件的任意位置

(1)java文件模型

在硬盘上的文件是byte byte byte存储的,是数据的集合

(2)打开文件

有两种模式”rw”(读写) “r” (只读)

RandomAccessFile raf = new RandomAccessFile(file,”rw”)

文件指针,打开文件时指针在开头 pointer=0

(3)写方法

raf.write(int)🡪只写一个字节(后8位),同时指针指向下一个字节的位置

(4)读方法

int b = raf.read()🡪读一个字节

(5)文件读写完成以后一定要关闭(Oracle官方说明,否则可能会发生未知错误)

4-1字节流之文件输入流FileInputStream-1

IO流(输入流、输出流)

字节流、字符流

1. 字节流
2. InputStream、OutputStream(抽象类)

InputStream抽象了应用程序读取数据的方式

OutputStream抽象了应用程序写出数据方式

1. EOF =End 读到-1就读到结尾
2. 输入流基本方法（键盘是输入文件，用来读）

int b = in.read();//返回值为整数。读取一个字节无符号填充到int的低八位。-1是EOF

in.read(byte[] buf);//读取数据填充到字节数组buf

in.read(byte[] buf,int start,int size);//读取数据到字节数组buf。从buf的start位置开始存放size长度的数据

1. 输出流基本方法

out.write(int b); //写出一个byte到流，b的八8位

out.write(byte[] buf); //将buf字节数组都写入到流

out.write(byte[] buf,int start,int size); //字节数组从start位置开始写size擦很难过度的字节到流

1. FileInputStream🡪具体实现的在文件上读取数据
2. FileOutputStream🡪实现了向文件中写出byte数据的方法
3. DataOutputStream/DataInputStream(过滤流)

对”流”功能的拓展，可以更加方便的读取int, long, 字符等类型

DataOutputStream

writeInt()/writeDouble()/writeUTF()

8) BufferedInputStream & BufferedOutputStream(过滤流)

这两个流类为IO提供了带缓冲区的操作，一般打开文件进行写入或读取操作时，都会加上缓冲，这种模式提高了IO的性能

从应用程序中把输入放入文件，相当于将一缸水导入另一个缸中,与所用时间无关

FileOutputStream🡪write()方法相当于一滴一滴地把水”转移”过去

DataOutputStream🡪writeXxx()方法，相当于一瓢一瓢把水”转移”过去

BufferedOutputStream🡪write()方法更方便，相当于一瓢一瓢先放入桶中，再从桶中倒入另一个缸中，性能更好

1. 字符流
2. 编码问题
3. 认识文本和文本文件

java的文本(char)是16位无符号整数，是字符的unicode编码（双字节编码）

文件是byte byte byte。。。的数据序列

文本文件是文本（char）序列按照某种编码方案（utf-9,utf-16e,gbk）序列化为byte的存储结果

3）字符流（Reader Writer）🡪操作的是文本文件

字符的处理，一次处理一个字符。字符的底层仍然是基本的字节序列

字符流的基本实现

InputStreamReader 完成byte流解析为char流，按照编码解析

OutputStreamWriter 提供char流到byte流，按照编码处理

FileReader/FileWriter

字符流的过滤器

BufferedReader🡪 readLine 一次读一行

BufferedWriter/PrintWriter🡪一次写一行

1. 对象的序列化、反序列化
2. 对象序列化，就是将Object转换成byte序列，反之叫对象的反序列化
3. 序列化流（ObjectOutputStream）是字节的过滤流🡪writeObject

反序列化流（ObjectInputStream）🡪readObject

1. 序列化接口（Serializable）

对象必须实现序列化接口，才能进行序列化，否则将出现异常

这个接口没有任何方法，只是一个标准

1. transient关键字

**private** **void** writeObject(java.io.ObjectOutputStream s)

**throws** java.io.IOException

**private** **void** readObject(java.io.ObjectInputStream s)

**throws** java.io.IOException, ClassNotFoundException

分析ArrayList源码中序列化和反序列化的问题

1. 序列化中 子类和父类构造函数的调用问题

若父类实现了序列化接口,子类就不用实现了

对子类对象进行反序列化操作时，如果其父类没有实现反序列化接口那么其父类的构造函数就会被调用