IO流总结

概念

\* IO流用来处理设备之间的数据传输

\* Java对数据的操作是通过流的方式

\* Java用于操作流的类都在IO包中

\* 流按流向分为两种：输入流，输出流。

\* 流按操作类型分为两种：

\* 字节流 : 字节流可以操作任何数据,因为在计算机中任何数据都是以字节的形式存储的

\* 字符流 : 字符流只能操作纯字符数据，比较方便。

字节流

|--InputStream(字节输入流,字节流的抽象父类,抽象类不能直接创建对象,必须使用子类)

|--FileInputStream(文件字节输入流,就是以字节的形式读取文件内容)

|--read()每次读取一个字节(byte),但是返回值类型,为int类型,防止出现文件提前结束

|--read(byte[] b)每次读取多个字节,将内容放入字节数组b中

|--BufferedInputStream(字节输入缓冲流,其实就是一个包装类,创建对象时要接收一个InputStream,内部封装了一个1024\*8的字节数组)

|--使用BufferedInputStream就不用再定义数组了

|--OutputStream(字节输出流,字节流的抽象父类)

|--FileOutputStream(文件字节输出流,就是以字节的形式将内容写入到文件中)

|--write(int b)一次写出一个字节

|--write(byte[] b)将字节数组b里的内容写入到文件中

|--write(byte[] b, int off, int len)将字节数组b里的内容,从索引为off的位置开始,总共写len个

|--BufferedOutputStream(字节输出缓冲流,其实就是一个包装类,创建对象时要接收一个OutputStream,内部封装了一个1024\*8的字节数组)

字符流

|--Reader(字符输入流,字符流的抽象父类,抽象类不能直接创建对象,必须使用子类)

|--FileReader(文件字符输入流,就是以字符的形式读取文件中的内容)

|--read()每次读取一个字符,即可以按照字符大小读取,返回值类型提升为int类型

|--read(char[] b)每次读取多个字符,将内容放入字符数组b中

|--BufferedReader(字符输入缓冲流,其实就是一个包装类,创建对象时要接收一个Reader,内部封装了一个1024\*8的字符数组)

|--使用BufferedReader和BufferedWriter就不用再定义数组了

|--readLine()可以读取一行字符(不包含换行符号)

|--Writer(字符输出流,字符流的抽象父类)

|--FileWriter(文件字符输出流,就是以字符的形式将内容写入到文件中)

|--write('a')/write("aaaa")可以写入单个字符或者写入字符串

|--write(char[] b, int off, int len)将字符数组b里的内容,从所以为off的位置开始,总共写len个

|--BufferedWriter(字符输出缓冲流,其实就是一个包装类,创建对象时要接收一个Writer,内部封装了一个1024\*8的字符数组)

|--newLine()可以输出一个跨平台的换行符号"\r\n"

转换流(使用指定的码表读写字符)

|--InputStreamReader字节输入流到字符输入流的桥梁

|--OutputStreamWriter字符输出流到字节输出流的桥梁

其它流

|--ByteArrayOutputStream字节数组内存输出流

|--是将文件或键盘录入的数据写入到内存中的字节数组

|--RandomAccessFile随机读写访问流

|--该流同时具有读和写的功能

|--seek(long pos)设置读取和写入的位置

|--对象操作流

|--ObjectOutputStream将对象写入到文件,要求对象所属的类实现Serializable接口,该接口里面没有内容,只是一个标记

|--writeObject()将对象写入文件

|--ObjectInputStream将上面存入到文件中的对象读取出来,因为存的时候所有对象是按照Object存入,所以取出来要进行强制类型转换

|--readObject()从文件读取一个对象出来

|--打印流

|--PrintStream有很多的print和println方法，直接以字符串形式打印数据

|--PrintWriter具有自动刷新的功能

|--标准输入输出流

|--System.in是InputStream, 标准输入流, 默认可以从键盘输入读取字节数据

|--System.setIn(InputStream)修改输入流

|--System.out是PrintStream, 标准输出流, 默认可以向Console中输出字符和字节数据

|--Properties是Hashtable的子类,是一个Map,存放的是键值对,一般用于项目的配置文件,配置文件中键和值的类型都为String类型

|--setProperty(String key,String value)向Properties中存入键值对

|--getProperty(String key)根据键获取值

|--stringPropertyNames()以String的形式获取所有的键,并封装成Set对象,之后进行遍历

|--load(InputStream inStream) 将字节输入流inStream所关联的文件中的值加载到Properties对象中

|--store(OutputStream out, String comments) 将Properties对象中的内容写入到out流所关联的文件中,comments为说明信息

flush用来刷新缓冲区的,刷新后可以再次写出,即flush后流仍然可以使用

close用来关闭流释放资源的的,如果是带缓冲区的流对象的close()方法,不但会关闭流,还会再关闭流之前刷新缓冲区,关闭后不能再写出,即不能再使用该流

IO流的标准异常处理