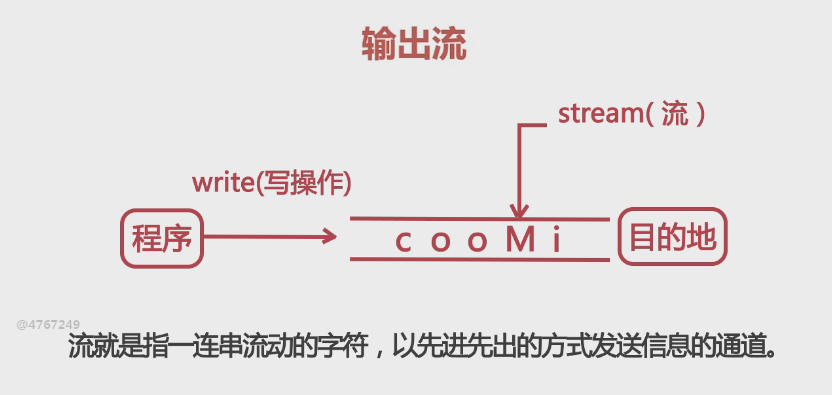
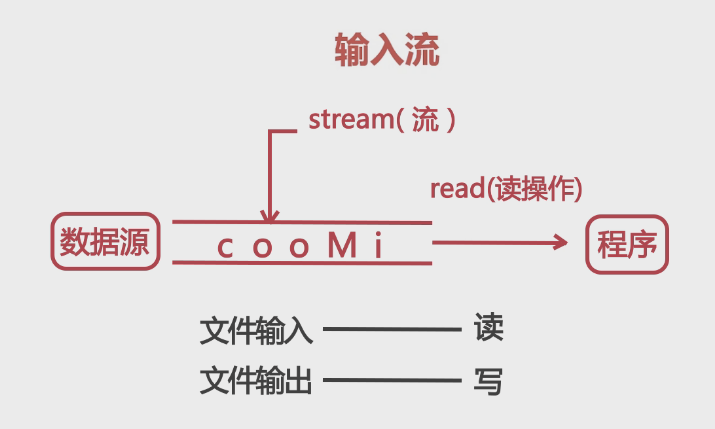
System.out.println(“hello world”);



Scanner sc = new Scanner(Systtem.in);



主要内容：

一：File类的使用

二：字节流

三：字符流

四：对象的序列化与反序列化

一：File类的使用

1.

在java中使用java.io.File类对文件进行操作

**Windows中分隔符为\ linux的分隔符为/**

**File f1 = new File(“c:\\imooc\\io\\score.txt”);**

**要用**[**\\作为分隔符**](file:///\\作为分隔符) **单个\进行转义**

2.

查看API的File类 有4个构造方法：

File(File parent, String child)

File(String pathname)

File(String parent, String pathname)

File(URI uri)

File类的常用方法：boolean carRead() , boolean canWrite(), boolean exits(),

String getName(), Boolean isDriectory(), Boolean isFile(), long lastModified(),

Boolean isHiden(), Boolean mkdir(), Boolean mkdirs().

3.

exits() 判断目录或文件 是否存在

isDriectory() 判断是否是目录

isFIle() 判断是否是文件

mkdir() **只能创建一级目录** （不能超过两级）

mkdirs() **能创建多级目录**

createNewFile() 创建文件 （如果不写扩展名，创建出来的是没有扩展名的文件）

4.

绝对路径和相对路径

绝对路径：从盘符开始的路径

相对路径：从当前路径开始的路径（从当前类的路径开始路径）

相对路径：从当前路径开始的路径。 通过 ..\\ 返回上级目录

**Eclipse中相对路径是 工程所在的路径。**

**isAbsolute()**  返回true 则是绝对路径 返回false 则是相对路径

**getAbsolutePath() 获取绝对路径**

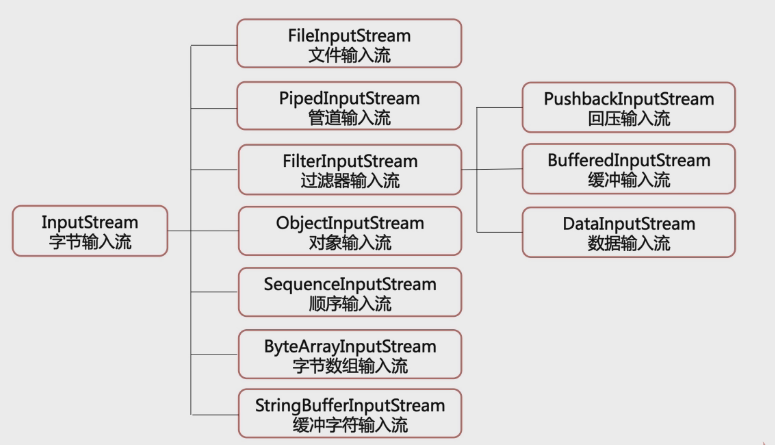
**getPath() 获取相对路径**

二．字节流

1.

字节输入流 InputStream

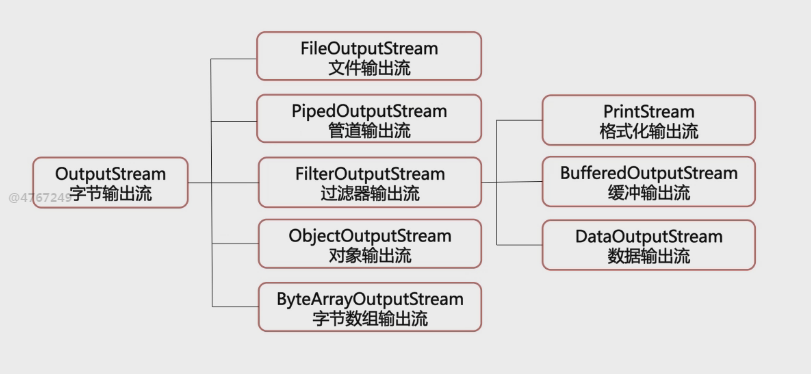
字节输出流 OutputStream



主要介绍：FileInputStream 文件输入流（从文件中读取数据）

BufferedInputStream 缓冲输入流 （有缓冲区，提高读数据的速度）

ObjectInputStream 对象输入流



主要介绍：FileOutputStream 文件输出流

BufferedOutputStream 缓冲输出流

ObjectOutputStream 对象输出流

2.

FileInputStream 文件输入流（从文件系统中的某个文件中获得输入字节）

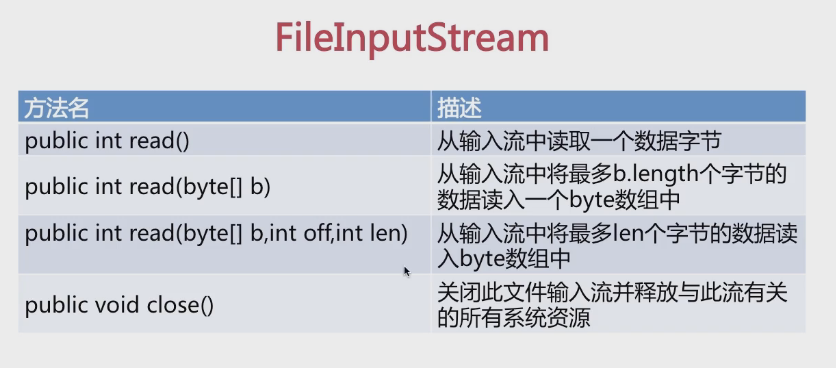
（用于读取诸如图像数据的之类的原始字节流） 对于字符要用 字符流

构造：FileInputStream(File file)

FileInputStream(String name)

FileInputStream(FileDescriptor fdObj) 用文件描述符创建对象

例：FileInputStream(FileDescriptor.in)



**read() 方法如果返回值为-1，表示已经 读取到文件末尾。**

**read(byte[] b) 将数据读取到数组中**

**read(byte[] b, int off, int len) off表示偏移量（一般为0，大于0，表示从b数组的第几个位置开始存放数据）， len表示从文件中读取的字节大小**

**close() 关闭流**

3.

FileOutputStream 文件输出流

构造方法：

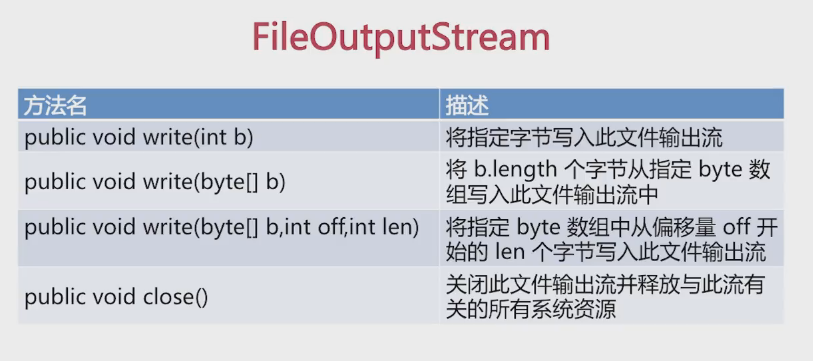
FileOutputStream(File file)

FileOutputStream(String name)

FileOutputStream(File file, boolean append) 后面为true表示在文件最后追加，否则就是覆盖

FileOutputStream(String name, Boolean append) 后面为true表示在文件最后追加，否则就是覆盖

常用方法：



例：

fis = **new** FileInputStream("hello.jpg");

fos = **new** FileOutputStream("hellocopy2.jpg");

**byte**[] b = **new** **byte**[1024];

**int** n = 0;

**while**((n = fis.read(b)) != -1){

fos.write(b, 0, n);

}

// write(b,0,n)该方法 让复制后的文件大小 不变

//（当最后一次读取的大小不足以补充满b的大小时,就按照实际的存储）

4. 缓冲流（读取速度更快 从内存中读取）

缓冲输入流：BufferedInputStream

缓冲输出流：BufferedOutputStream

缓冲流不能直接读取文件中的数据， 要和输入输出流配合使用

缓冲输入流构造：

BufferedInputStream(InputStream in) 参数可以是任何InputStream 的子类

BufferedInputStream(InputStream in， int size) 创建指定大小的缓冲流

常用方法：和FileInputStream类似

read()

read(byte[] b, int off, int len)

缓冲输出流构造：

BufferedOutputStream(OutputStream out) 参数可以是任何OutputStream 的子类

BufferedOutputStream(OutputStream out, int size) 创建指定大小的缓冲流

常用方法：

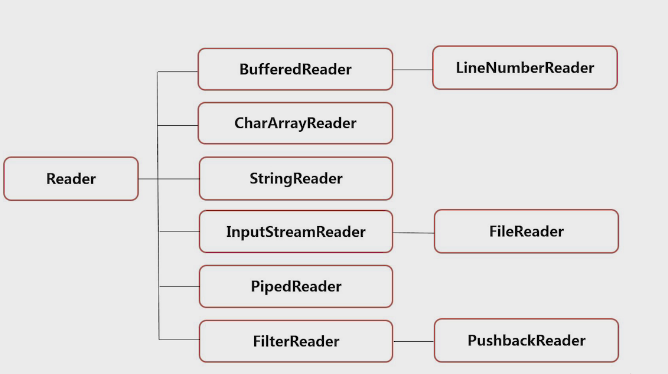
flush() 清空缓冲区（缓冲区不满时，强制清空）。缓冲区满了自动执行写操作。缓冲区不满，进行强制清空。

write(byte[] b)

write(byte[] b, int off, int len)

三．字符输入流：Reader

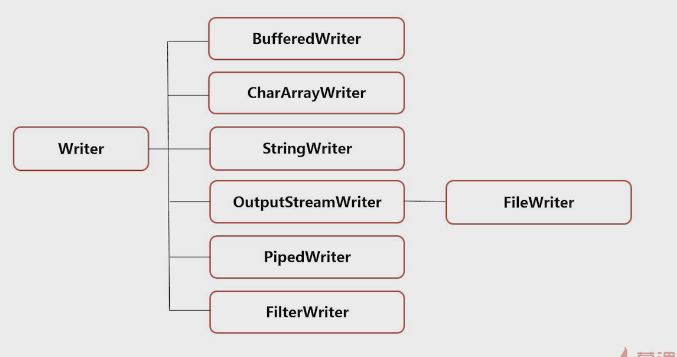
字符输出流：Writer



主要介绍：BufferedReader 缓冲输入流

**InputStreamReader 字节流和字符流转换的流（很重要）**

FileReader 文件输入流



主要介绍：BufferedWriter 缓冲输出流

**OutputStreamWriter 字节流和字符流转换的流（重要）**

FileWriter 文件输出流

字节字符转换流 (字节和字符过度的桥梁)

InputStreamRead

OutputStreamWriter

主要构造：InputStreamRead(InputStream in)

InputStreamRead(InputStream in, String charsetName)

方法： void close(), String getEncoding(), int read(),

int read(char[] cbuf, int offset, int length), boolean ready()

主要构造：OutputStreamWriter(OutputStream out)

OutputStreamWriter(OutputStream out, String charsetName)

方法： void close(), String getEncoding(), void flush()

void write(char[] cbuf, int offset, int length),

void write(String str, int offset, int length),

void write(int c)

2.

缓冲流：BufferedWrite

BufferedReader

构造：BufferedReader(Reader in)

BufferedReader(Reader in, int size) 可以指定缓冲区的大小

常用方法：int read(),int read(char[] cbuf, int off, int len), String readLine()

构造：BufferedWrite(Write out)

BufferedWrite(Write out, int size)

常用方法：void close(), void flush(), void newLine()

四．对象序列化和反序列化

1.步骤：

创建一个类，继承Serializable接口（只有继承这个接口，类的对象才能读写）

创建对象

将对象写入文件

从文件读取对象信息

涉及到2个类：

ObjectInputStream 对象输入流

ObjectOutputStream 对象输出流

构造方法：

ObjectOutputStream() ObjectInputStream()

ObjectOutputStream(OutputStream out) ObjectInputStream(InputStream in)

常用方法：

writeObject(Object obj) Object readObject()