**IO流**

IO流的分类：IO用于在设备间进行数据传输的操作

1. 按照数据流的方向不同可以分为：

输入流 读取数据

文件

程序

输出流 写出数据

文件

程序

2. 按照实现功能不同可以分为：节点流和处理流。

节点流：直接与数据源相连，读入或读出。直接使用节点流，读写不方便，为了更快的读写文件，才有了处理流。

处理流：与节点流一块使用，在节点流的基础上，再套接一层，套接在节点流上的就是处理流。

3. 按照处理数据单位不同可以分为：字节流和字符流

输入和输出都是从程序的角度来说的。字节流：一次读入或读出是8位二进制。字符流：一次读入或读出是16位二进制。字节流和字 符流的原理是相同的，只不过处理的单位不同而已。后缀是Stream是字节流，而后缀是Reader，Writer是字符流。

字节流

字节输入流 读取数据 InputStream

字节输出流 写出数据 OutputStream

字符流

字符输入流 读取数据 Reader

字符输出流 写出数据 Writer

注意：一般我们在探讨IO流的时候，如果没有明确说明按哪种分类来说，默认情况下是按照数据类型来分的。

**File类：**

我们要想实现IO的操作，就必须知道硬盘上文件的表现形式。而Java就提供了一个类File供我们使用

File:文件和目录(文件夹)路径名的抽象表示形式

构造方法：

File(String pathname)：根据一个路径得到File对象

File(String parent, String child):根据一个目录和一个子文件/目录得到File对象

File(File parent, String child):根据一个父File对象和一个子文件/目录得到File对象

创建功能：

public boolean createNewFile():创建文件 如果存在这样的文件，就不创建了

public boolean mkdir():创建文件夹 如果存在这样的文件夹，就不创建了

public boolean mkdirs():创建文件夹,如果父文件夹不存在，会帮你创建出来

删除功能:

public boolean delete()

注意：如果你创建文件或者文件夹忘了写盘符路径，那么，默认在项目路径下。Java中的删除不走回收站。要删除一个文件夹，请注意该文件夹内不能包含文件或者文件夹

重命名功能:

public boolean renameTo(File dest)

如果路径名相同，就是改名。如果路径名不同，就是改名并剪切

判断功能:

public boolean isDirectory():判断是否是目录

public boolean isFile():判断是否是文件

public boolean exists():判断是否存在

public boolean canRead():判断是否可读

public boolean canWrite():判断是否可写

public boolean isHidden():判断是否隐藏

获取功能：

public String getAbsolutePath()：获取绝对路径

public String getPath():获取相对路径

public String getName():获取名称

public long length():获取长度。字节数

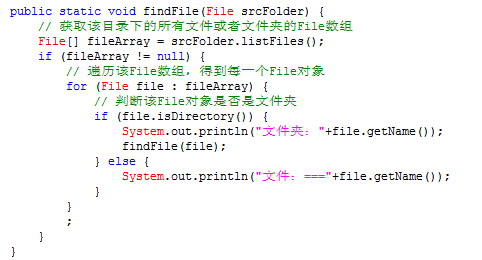
public long lastModified():获取最后一次的修改时间，毫秒值

public String[] list():获取指定目录下的所有文件或者文件夹的名称数组

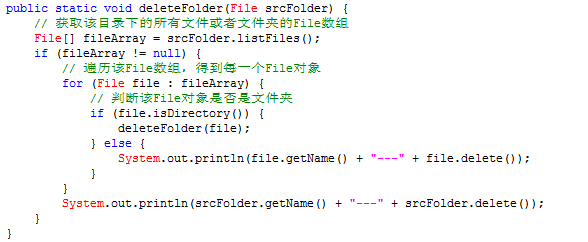
public File[] listFiles():获取指定目录下的所有文件或者文件夹的File数组

File类编码

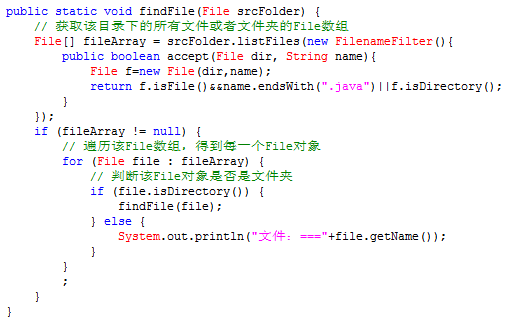
1. 遍历指定目录以及子目录下的文件



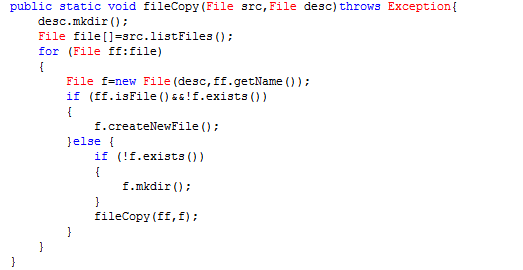
1. 递归删除带内容的目录



1. 文件过滤：显示目录下的所有java文件

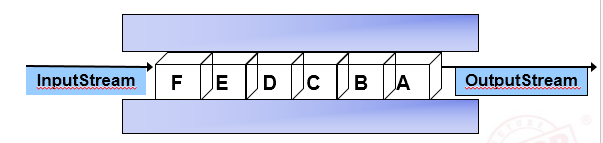


1. 将指定目录下以及子目录下的文件及目录拷贝到另外一个目录

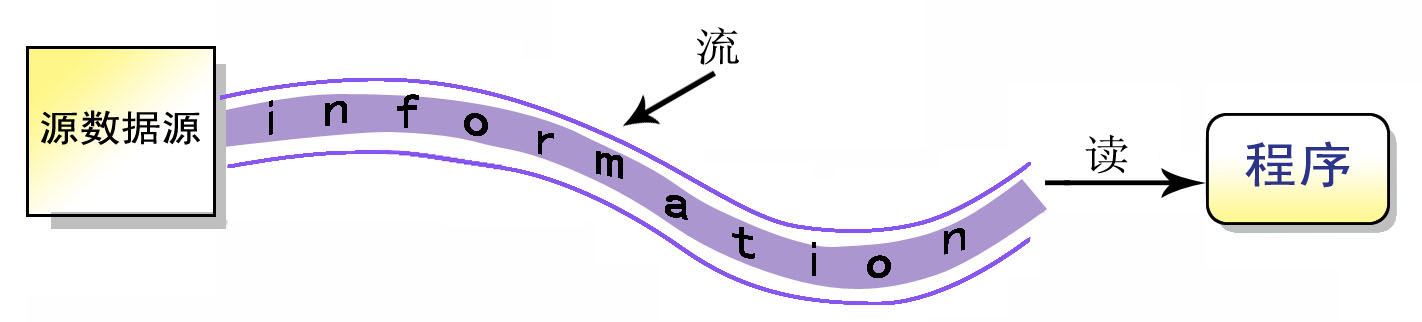


**文件流：**

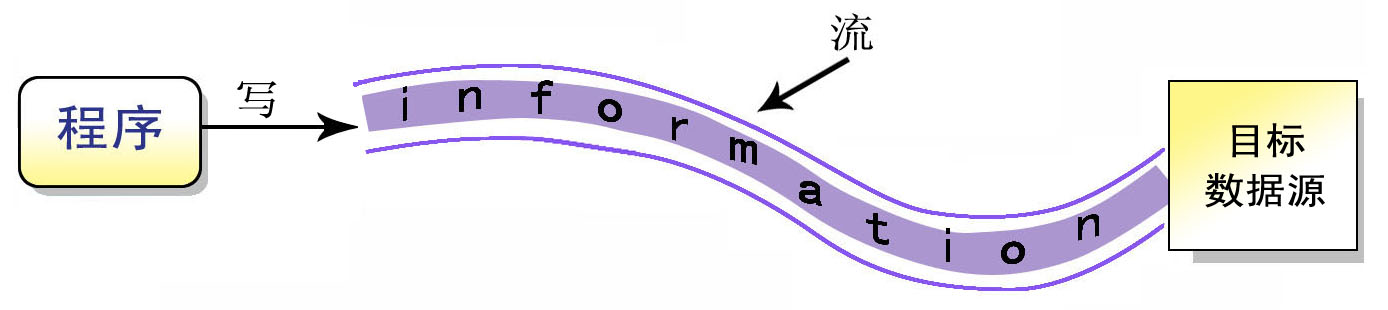
流是指一连串流动的字符，是以先进先出方式发送晓得通道。



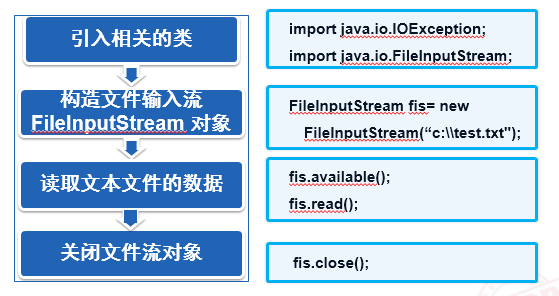
输入流：



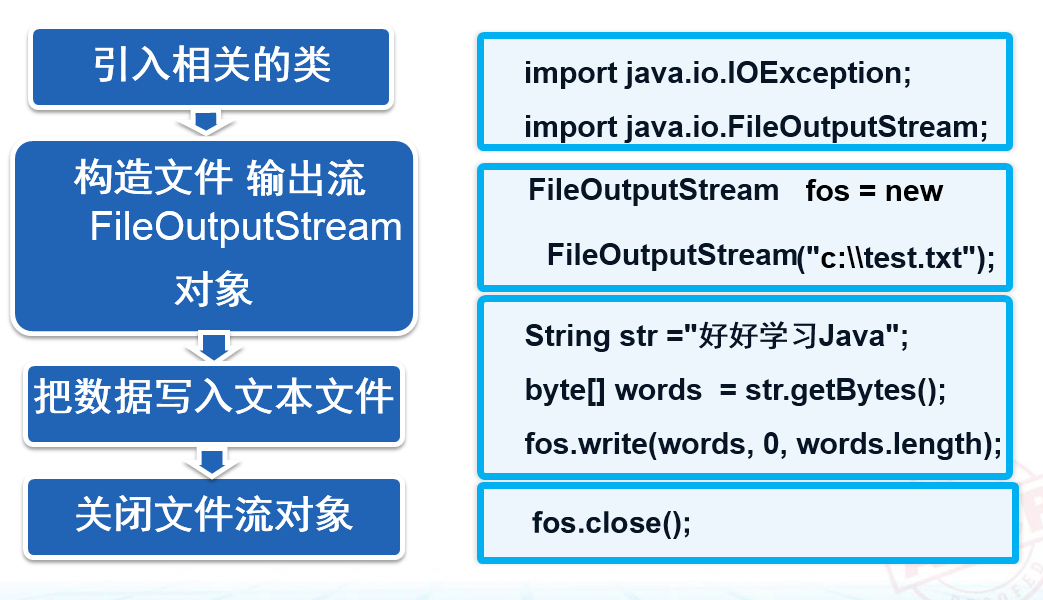
输出流：



使用FileInputStream读文本文件：



使用FileOutputStream读文本文件：



使用FileReader读文本文件：



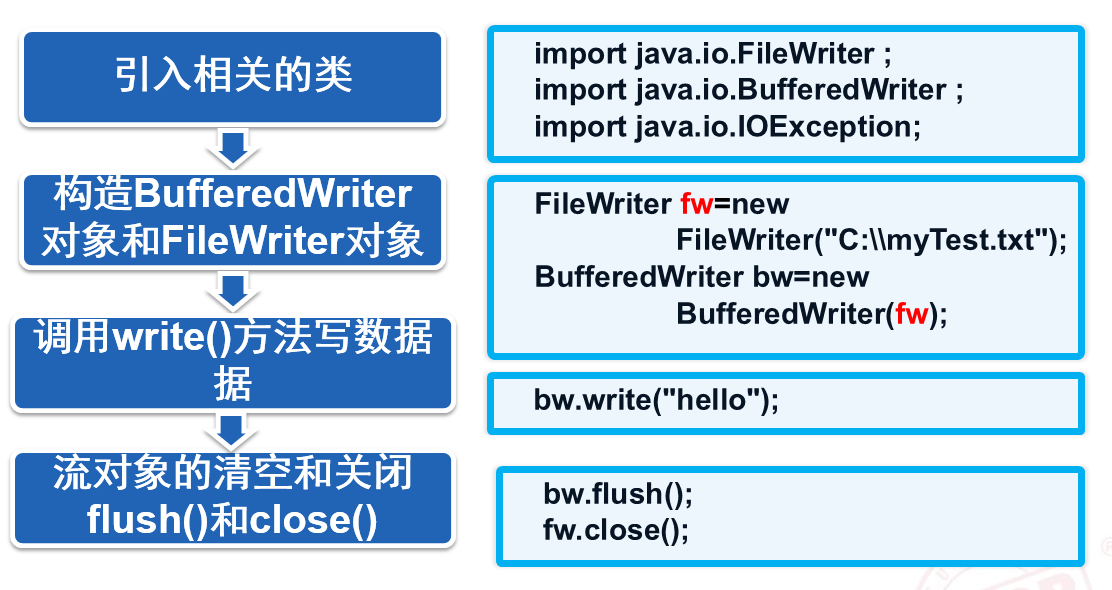
使用BufferedReader读文本文件:



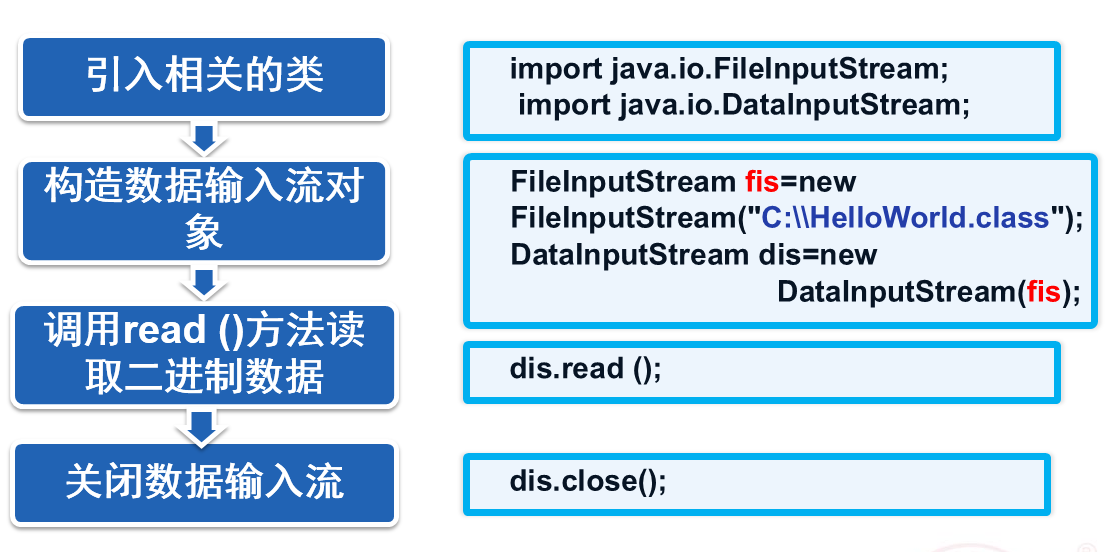
使用FileWriter 写文本文件：



使用BufferedWriter写文本文件:



使用DataInputStream 读二进制文件:



使用DataInputStream 读二进制文件:

