# Java IO：面向流、阻塞IO

## Java.io包下主要有四种类：

1. 基于字节操作的IO：InputStream、OutputStream
2. 基于字符操作的IO：Writer、Reader
3. 基于磁盘操作的IO：File
4. 基于网络操作的IO：Socket

## 字节输入流-InputStream

方法：read 读入流中数据

available 返回流中还可以读取的字节数目

clous 关闭流

实现类： FileInputStream 对一个文件进行字节读取操作

ByteArrayInputStream 将字节读入字节数组

## 字节输出流-OutputStream

方法：write 数据写到流中

flush 剩余部分全部写入，然后清空缓冲区

close 关闭流

实现类： ByteArrayOutputStream 将数据写入字节数组

FileOutputStream 将数据写入文件

## 字符输入流-Reader

与InputStream类似

## 字符输出流-Writer

与OutputStream类似

## 转换流

InputStreamReader 从流中读取字节，然后转换为字符

OutputStreamWriter

## 字符串流

StringReader

StringWriter

## 缓冲流：性能更好

BufferedInputStream / BufferedOutputStream

BufferReader / BufferWriter

## 随机访问：RandomAccessFIle

可以读、写文件

# NIO：Non-Blocking IO

NIO是面向缓冲区、基于通道的IO、非阻塞IO，需要线程少。

## 流与缓冲区

面向流：即一次IO中，需要向流中写入或者从流中读取一定的数据，这些数据不会被缓存，只能等待读取完毕。因此数据不能在流中被操作。

面向缓冲区：一次IO中，数据会先被放置在缓冲区内，数据在缓冲区中可以被操作。

## 阻塞与非阻塞

阻塞IO操作时，一个线程只能等待操作完毕，这段时间不能进行其他任务。

非阻塞IO操作时，线程创建缓冲区，缓冲区和通道交互数据，交互时线程不阻塞。

## 选择器

Java IO没有选择器，每个线程负责一个IO。

NIO有选择器，允许一个线程通过选择器监视多个管道。由于操作系统切换线程消耗大，所以选择器可以提高效率。

## NIO组件

## Channel：IO从channel开始

通道介于程序与实体之间，因此有许多类型的。通道可以是双向的、非阻塞的。从channel读取数据写入到buffer，从buffer读取数据写入channel。

常用的通道

FileChannel 读写文件 / SocketChannel TCP通信

ServerSocketChannel 监听TCP连接 DatagramChannel UDP通信

## Buffer：一块内存

NIO使用的缓冲区是经过封装的byte[]，有ByteBuffer/CharBuffer/

IntBuffer等，类型不同读写缓冲区时操作的单位也不同。

Buffer有三个关键属性，capacity容量、position位置、limit边界

## Selector

Selector用于采集各个通道的状态。因此需要将通道绑定到选择器上，并且设置好关心的状态，通过调用select()方法采集状态，没有合适的则挂起，让出CPU。

通道有四个可以监听的状态：

Accept：有可以接受的连接

Connect：连接成功

Read：可以读数据

Write：可以写数据

## NIO与IO适用情况：

连接数目多，每个连接数据量小，适用NIO。

连接数目少，每个连接数据量大，适用普通IO。