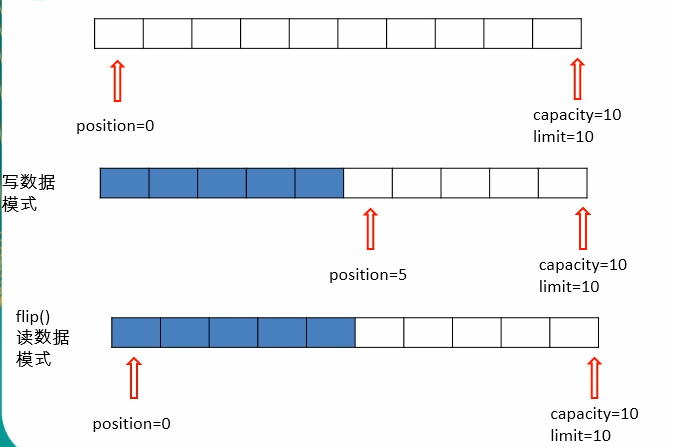
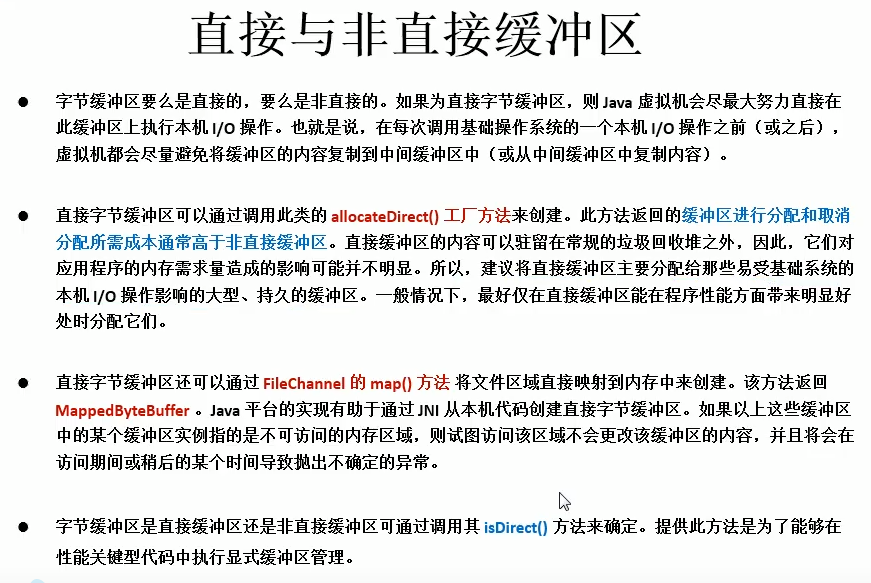
## TCP-NIO-epoll-netty

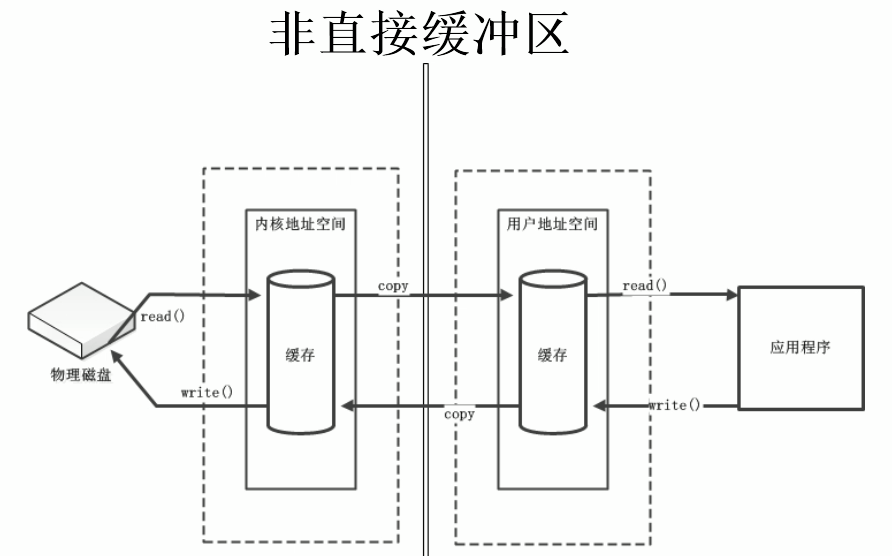
1. 传输控制层的协议：UDP,TCP
2. 传输控制层：面向连接，可靠的传输
3. TCP三次握手
   1. 客户端向服务器端发送syn包：测试能否连接到服务器
   2. 服务器端收到syn包后回应一个syn-apk包：说明客户端到服务器端是可以连通的
   3. 客户端收到syn-apk包后回应一个ack包：说明服务器端到客户端是可以连通的
4. Socket：套接字
5. 四次挥手
   1. 客户端发送一个fin数据包到服务器说明想要断开连接
   2. 服务器端收到fin数据包后回送fin-ack数据跑，表明收到了要断开连接的请求
   3. 服务器发送一个fin数据包到客户端，表名服务器也想断开连接了
   4. 客户端收到服务区发送的fin之后，回一个ack数据包，说明收到了服务区发送的fin数据包
6. 数据传输时，数据包的长度的大小最大是1460
7. 三次握手到四次挥手不可分割
8. Java当中什么是线程：java中的线程就是通过调用内核的系统调用然后得到操作系统中的一个轻量级的进程（有进程号）
9. Nio：java的new io
10. 什么是nio：指的就是不堵塞，非阻塞io
11. 如果程序自己读取io，那么这个io模型，无论BIO，NIO，多路复用器：同步IO模型
12. IO 和NIO的区别
    1. Io面向流（Stream Oriented） Nio面向缓冲区（Buffer Oriented）
    2. Io阻塞io（Blocking Io） NIO非阻塞io（Non Blocking io）
    3. Nio选择器：selectors
13. 缓冲区中的四个核心属性
    1. Capacity：容量，表示缓冲区中最大存储数据的容量。一旦声明不能改变
    2. Limit：界限，表示缓冲区可以操作数据的大小。（limit后数据不能进行读写）
    3. Position：位置，表示缓冲区中正在操作数据的位置
    4. Mark：标记，标记当前position的位置，可以通过reset（）恢复到mark位置
    5. 0 <= mark <= position <= limit <= capacity
14. 缓冲流程图



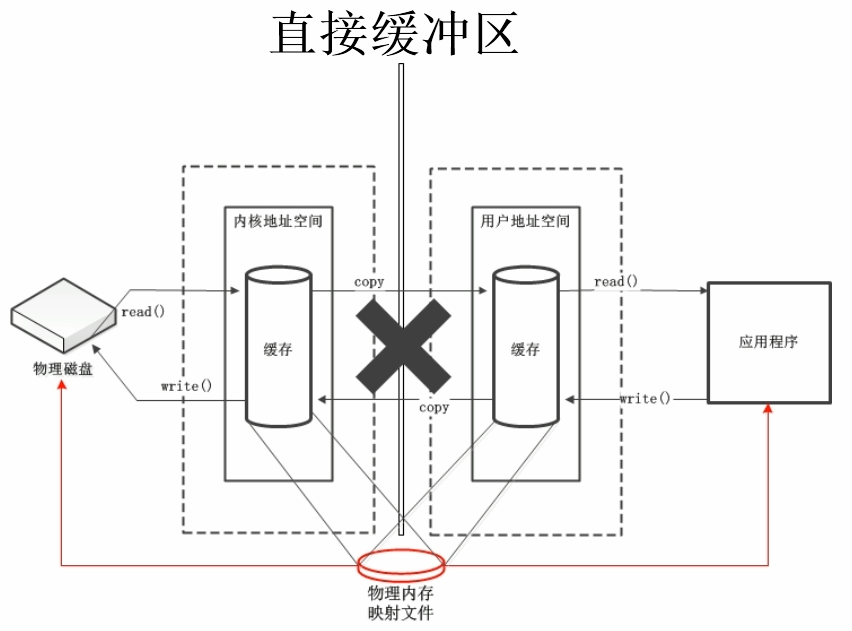
1. 直接缓冲区和非直接缓冲区
   1. 非直接缓冲区：通过allocate（） 方法分配缓冲区，将缓冲区建立在jvm的内存中
   2. 直接缓冲区：通过allocateDirect（）方法分配直接缓冲区，将缓冲区建立在物理内存中



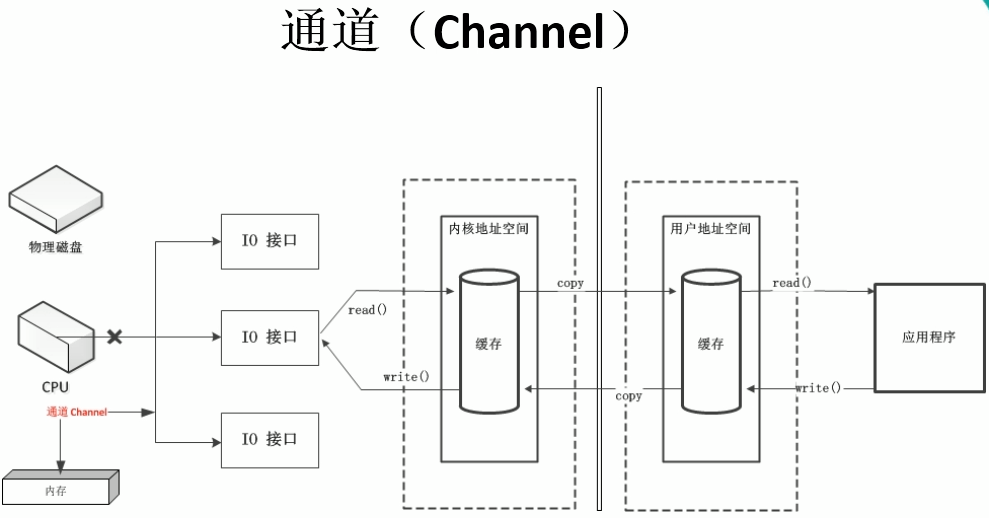
1. 非直接缓冲区



1. 直接缓冲区



1. 通道（channel）



1. 通道的主要实现类
   1. FileChannel
   2. SocketChannel
   3. ServerSocketChannel
   4. DatagramSocket
2. 字符集
   1. 编码：字符串 -> 字节数组
   2. 解码：字节数组 -> 字符串