http://wiki.jikexueyuan.com/project/java-nio-zh/java-nio-channel-to-channel-transfers.html

在Java NIO中如果一个channel是FileChannel类型的，那么他可以直接把数据传输到另一个channel。逐个特性得益于FileChannel包含的transferTo和transferFrom两个方法。

transferFrom()

FileChannel.transferFrom方法把数据从通道源传输到FileChannel：

```
RandomAccessFile fromFile = new
RandomAccessFile("fromFile.txt", "rw");
FileChannel fromChannel = fromFile.getChannel();
RandomAccessFile toFile = new RandomAccessFile("toFile.txt",
"rw");
FileChannel toChannel = toFile.getChannel();
long position = 0;
long count = fromChannel.size();
toChannel.transferFrom(fromChannel, position, count);
```

transferFrom的参数position和count表示目标文件的写入位置和最多写入的数据量。如果通道源的数据小于count那么就传实际有的数据量。 另外，有些SocketChannel的实现在传输时只会传输哪些处于就绪状态的数据，即使SocketChannel后续会有更多可用数据。因此，这个传输过程可能不会传输整个的数据。

transferTo()

transferTo方法把FileChannel数据传输到另一个channel,下面是案例：

```
RandomAccessFile fromFile = new
RandomAccessFile("fromFile.txt", "rw");
FileChannel fromChannel = fromFile.getChannel();
```

```
RandomAccessFile toFile = new RandomAccessFile("toFile.txt",
"rw");
FileChannel toChannel = toFile.getChannel();
long position = 0;
long count = fromChannel.size();
fromChannel.transferTo(position, count, toChannel);
```

这段代码和之前介绍transfer时的代码非常相似，区别只在于调用方法的是哪个FileChannel.

SocketChannel的问题也存在与transferTo.SocketChannel的实现可能只在发送的buffer填充满后才发送，并结束。