

网络服务器

在第 10 题中，我使用了 `Socket` 作为通信手段。因此，对于题目提到的 `SocketChannel`，我又重新了解了一下。过程中，作为 `NIO` 的 `SocketChannel` 给我带来了许多学习上的障碍，这一点在以下的描述中也能体现。

实现一般的读写

理解如何创建一个 `SocketChannel` 并非难事，而第一道坎来自于其 `IO` 使用的 `ByteBuffer`。与 `Socket` 不同，`SocketChannel` 不是提供它自身的输入输出流，而是通过独立创建的 `ByteBuffer` 来进行数据传输。这要求使用者首先将信息转换为字节。

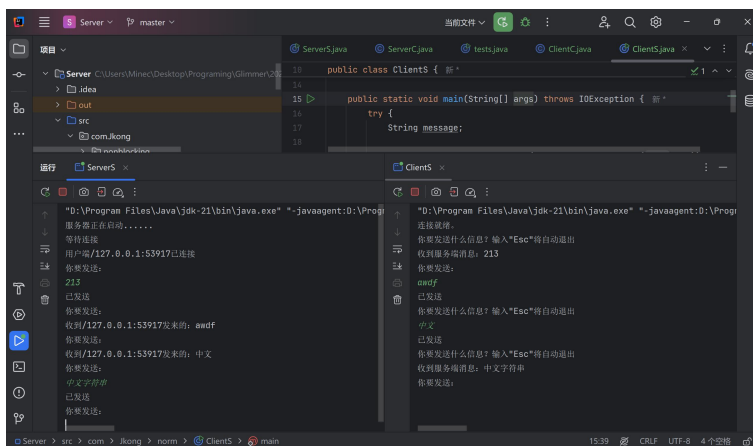
真正的难点是理解 `ByteBuffer` 的“指针”，即 `position`、`limit`、`mark`。在最初开始使用 `ByteBuffer` 时，我遇到的第一个问题就是消息后跟着一串乱码。通过检查发现，转字符串时，`ByteBuffer` 对应的数组中，没有意义的 `0` 也会一并被转换，这当然会导致乱码。而我花了一堆时间，终于意识到没有调用 `flip` 方法，从而 `limit` 仍位于数组末尾，会导致不应读取的字节被读取。

加上了 `flip` 方法后，英文自然是没有问题，但中文却仍是乱码。这次我很快意识到，中文的存储空间不是单个字节，而我采用每次读取一个字节的方式，当然会导致乱码。

通过观察，发现中文在 `ByteBuffer` 里会对应三个负的字节，于是我想，如果发现负字节，就一次读三个，否则仍一次读一个。结果当然仍是乱码。

使用 `wrap` 方法得到中文 `ByteBuffer` 数组，我发现除了负字节，数组尾部还会带上几个 `0`。这就使我意识到，中文在 `ByteBuffer` 的存储并不是“三个字节”这么简单。所以这促使我放弃了按字节的读取方式。

中间的曲折探索不妨省略，最后我采用的是使用 `utf8` 的 `encoder` 和 `decoder`，直接传输编解码后的 `ByteBuffer` 数组的方式，这很好地解决了上述的问题。最终效果如下图所示。对应的源码存放在 `Server\...\norm` 目录下。**ClientS** 是客户端发，**ClientC** 是客户端收，服务端同理。



实现一对多

上面的图片展示了服务端和客户端互相通信，实际上已经使用了双线程。主线程负责处理发送消息，而新建一个实现 Runnable 接口的类，用它新建一个用来处理接收消息的类。两者并行，互不干扰。

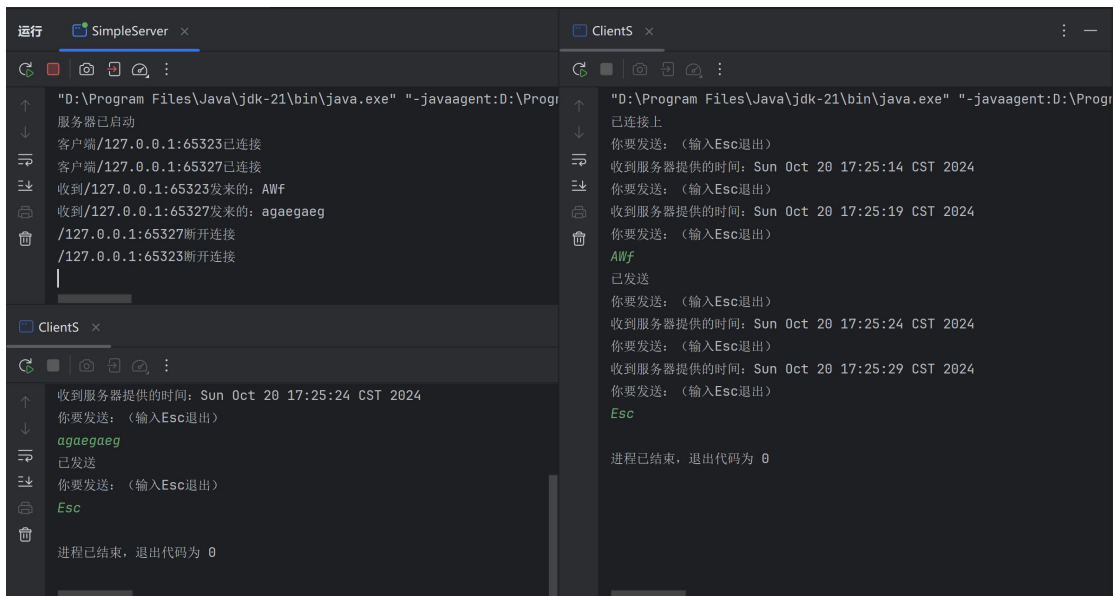
接下来，我又了解到，SocketChannel 常常设置成非阻塞模式并搭配 Selector “多路复用器”来使用，它可以让一个服务器用单个线程处理很多 Channel。

与上面相似，Selector 的创建、使用等基础工作并不难理解。我按照实例的思路，先将 Channel 注册到 Selector 上并标记关心的 IO 事件，然后调用 Selector 的 select 方法，再获取选择出的 Channel 的 Set，用 Iterator 处理其中的 SelectionKey。

在 ServerSocketChannel 的 accept 处理、SocketChannel 的 write 处理上，我都没有遇到明显的问题。但我发现，对于 read 事件，select 方法无法选出就绪的 Channel（一直堵塞），而使用 selectNow 方法就可以。这令我百思不得其解。按理说，select 的堵塞在它选择出就绪 Channel 后就会立即结束，但它没有。而 selectNow() > 0 的判断能通过（如下

图）又说明 Channel 确实已经就绪。`if (selector.selectNow() > 0)` 这个问题我暂时无法弄清，如果学长知道怎么回事，请多多指教。

源码放在 Server\...\nonblocking 目录下。由于服务端和客户端的代码很相似，我将注释都只写在服务端的代码里。以下是一对二的效果图。



```
运行 SimpleServer x
"D:\Program Files\Java\jdk-21\bin\java.exe" "-javaagent:D:\Progr
服务器已启动
客户端/127.0.0.1:65323已连接
客户端/127.0.0.1:65327已连接
收到/127.0.0.1:65327发来的: aagaegaeg
收到/127.0.0.1:65327发来的:
/127.0.0.1:65327断开连接
/127.0.0.1:65323断开连接

ClientS x
"D:\Program Files\Java\jdk-21\bin\java.exe" "-javaagent:D:\Progr
已连接上
你要发送: (输入Esc退出)
收到服务器提供的时间: Sun Oct 20 17:25:14 CST 2024
你要发送: (输入Esc退出)
收到服务器提供的时间: Sun Oct 20 17:25:19 CST 2024
你要发送: (输入Esc退出)
AWf
已发送
你要发送: (输入Esc退出)
收到服务器提供的时间: Sun Oct 20 17:25:24 CST 2024
你要发送: (输入Esc退出)
收到服务器提供的时间: Sun Oct 20 17:25:29 CST 2024
你要发送: (输入Esc退出)
Esc
进程已结束，退出代码为 0
```