1. Netty 是一个 基于 NIO 的 client-server(客户端服务器)框架，使用它可以快速简单地开发网
络应用程序。
2. 它极大地简化并优化了 TCP 和 UDP 套接字服务器等网络编程,并且性能以及安全性等很多
方面甚至都要更好。
3. 支持多种协议 如 FTP，SMTP，HTTP 以及各种二进制和基于文本的传统协议。
用官方的总结就是：Netty 成功地找到了一种在不妥协可维护性和性能的情况下实现易于开发，
性能，稳定性和灵活性的方法。
除了上面介绍的之外，很多开源项目比如我们常用的 Dubbo、RocketMQ、Elasticsearch、gRPC 等等都用到了 Netty

为什么要用 Netty?

统一的 API，支持多种传输类型，阻塞和非阻塞的。简单而强大的线程模型。
自带编解码器解决 TCP 粘包/拆包问题。
自带各种协议栈。
真正的无连接数据包套接字支持。
比直接使用 Java 核心 API 有更高的吞吐量、更低的延迟、更低的资源消耗和更少的内存复
制。
安全性不错，有完整的 SSL/TLS 以及 StartTLS 支持。
社区活跃
成熟稳定，经历了大型项目的使用和考验，而且很多开源项目都使用到了 Netty， 比如我们
经常接触的 Dubbo、RocketMQ 等等

Netty 应用场景了解么

1. 作为 RPC 框架的网络通信工具 ： 我们在分布式系统中，不同服务节点之间经常需要相互调
用，这个时候就需要 RPC 框架了。不同服务节点之间的通信是如何做的呢？可以使用 Netty
来做。比如我调用另外一个节点的方法的话，至少是要让对方知道我调用的是哪个类中的哪

个方法以及相关参数吧！
2. 实现一个自己的 HTTP 服务器 ：通过 Netty 我们可以自己实现一个简单的 HTTP 服务器，
这个大家应该不陌生。说到 HTTP 服务器的话，作为 Java 后端开发，我们一般使用 Tomcat 比多。一个最基本的 HTTP 服务器可要以处理常见的 HTTP Method 的请求，比如 POST 请求、GET 请求等等。
3. 实现一个即时通讯系统 ： 使用 Netty 我们可以实现一个可以聊天类似微信的即时通讯系
统，这方面的开源项目还蛮多的，可以自行去 Github 找一找。
4. **实现消息推送系统** ：市面上有很多消息推送系统都是基于 Netty 来做的。

Netty 核心组件有哪些？分别有什么作用

1.Channel

Channel 接口是 Netty 对网络操作抽象类，它除了包括基本的 I/O 操作，如 bind() 、 connect() 、 read() 、 write() 等。
比□常用的 Channel 接口实现类是 NioServerSocketChannel （服务端）和 NioSocketChannel （客
户端），这两个 Channel 可以和 BIO 编程模型中的 ServerSocket 以及 Socket 两个概念对应上。
Netty 的 Channel 接口所提供的 API，大大地降低了直接使用 Socket 类的复杂性。

2.EventLoop

EventLoop 的主要作用实际就是负责监听网络事件并调用事件处理器进行相关 I/O 操作的处理。那 Channel 和 EventLoop 直接有啥联系呢？Channel 为 Netty 网络操作(读写等操作)抽象类， EventLoop 负责处理注册到其上的 Channel处理 I/O 操作，两者配合参与 I/O 操作。

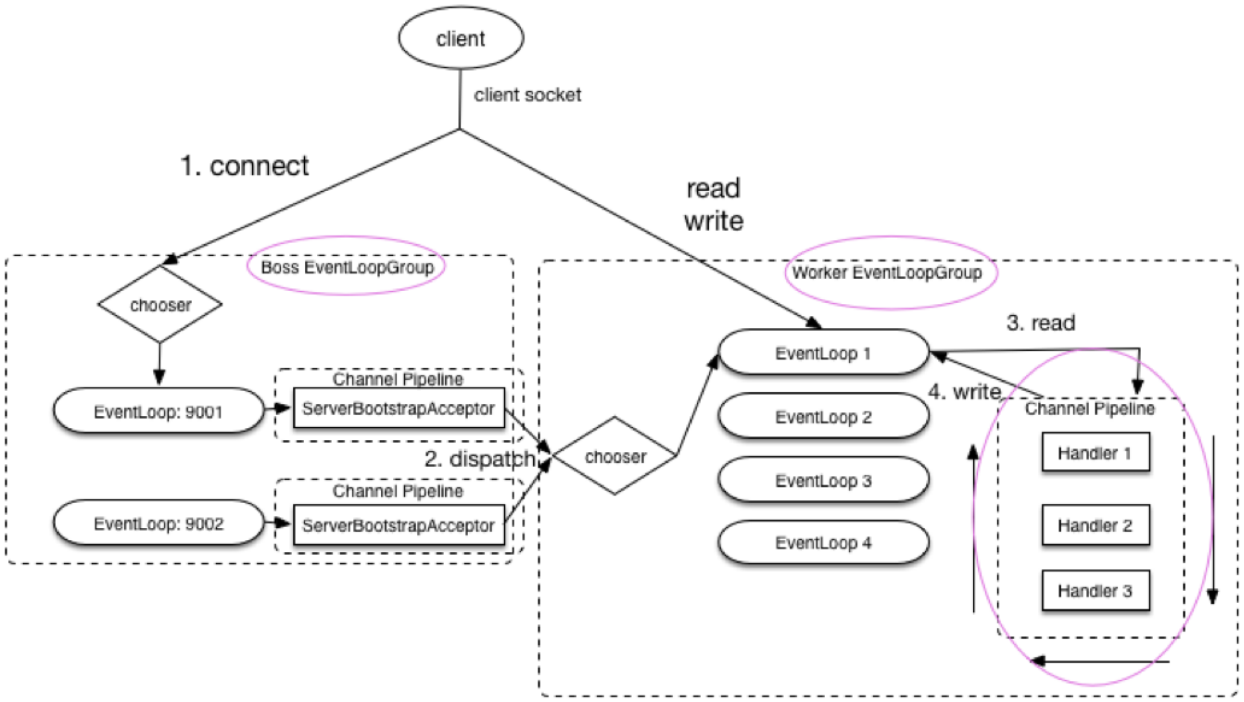
3.ChannelFuture

Netty 是异步非阻塞的，所有的 I/O 操作都为异步的。
因此，我们不能立刻得到操作是否执行成功，但是，你可以通过 ChannelFuture 接口的 addListener() 方法注册一个 ChannelFutureListener ，当操作执行成功或者失败时，监听就会自动
触发返回结果。并且，你还可以通过 ChannelFuture 的 channel() 方法获取关联的 Channel

4.ChannelHandler 和 ChannelPipeline

下面这段代码使用过 Netty 的小伙伴应该不会陌生，我们指定了序列化编解码器以及自定义的
ChannelHandler 处理消息。
ChannelHandler 是消息的具体处理器。他负责处理读写操作、客户端连接等事情。
ChannelPipeline 为 ChannelHandler 的链，提供了一个容器并定义了用于沿着链传播入站和出站
事件流的 API 。当 Channel 被创建时，它会被自动地分配到它专属的 ChannelPipeline 。
我们可以在 ChannelPipeline 上通过 addLast() 方法添加一个或者多个 ChannelHandler ，因为一
个数据或者事件可能会被多个 Handler 处理。当一个 ChannelHandler 处理完之后就将数据交给
下一个 ChannelHandler 。

EventloopGroup 了解么?和 EventLoop 啥关系?



EventLoopGroup 包含多个 EventLoop （每一个 EventLoop 通常内部包含一个线程），上面我
们已经说了 EventLoop 的主要作用实际就是负责监听网络事件并调用事件处理器进行相关 I/O
操作的处理。
并且 EventLoop 处理的 I/O 事件都将在它专有的 Thread 上被处理，即 Thread 和 EventLoop
属于 1 : 1 的关系，从而保证线程安全。

上图是一个服务端对 EventLoopGroup 使用的大致模块图，其中 Boss EventloopGroup 用于接收
连接， Worker EventloopGroup 用于具体的处理（消息的读写以及其他逻辑处理）。
从上图可以看出： 当客户端通过 connect 方法连接服务端时， bossGroup 处理客户端连接请
求。当客户端处理完成后，会将这个连接提交给 workerGroup 来处理，然后 workerGroup 负责
处理其 IO 相关操作

Netty 线程模型了解么

Reactor 模式基于事件驱动，采用多路复用将事件分发给相应的 Handler 处理，非常适合处
理海量 IO 的场景。
在 Netty 主要靠 NioEventLoopGroup 线程池来实现具体的线程模型的 。
我们实现服务端的时候，一般会初始化两个线程组：
1. bossGroup :接收连接。
2. workerGroup ： 负责具体的处理，交由对应的 Handler 处理。
下面我们来详细看一下 Netty 中的线程模型吧！
1.单线程模型 ：
一个线程需要执行处理所有的 accept 、 read 、 decode 、 process 、 encode 、 send 事件。对
于高负载、高并发，并且对性能要求比□高的场景不适用。