《Netty权威指南（第2版）》

图书基本信息

书名：《Netty权威指南（第2版）》

13位ISBN编号：9787121258013

出版时间：2015-4-20

作者：李林锋

页数：572

《Netty权威指南（第2版）》

内容概要

《Netty 权威指南（第2 版）》是异步非阻塞通信领域的经典之作，基于最新版本的Netty 5.0 编写，是国内首本深入介绍Netty 原理和架构的书籍，也是作者多年实战经验的总结和浓缩。内容不仅包含Java NIO入门知识、Netty 的基础功能开发指导、编解码框架定制等，还包括私有协议栈定制和开发、Netty 核心类库源码分析，以及Netty 的架构剖析。

《Netty 权威指南（第2 版）》适合架构师、设计师、软件开发工程师、测试人员以及其他对Java NIO 框架、Netty 感兴趣的相关人士阅读，通过《Netty 权威指南（第2 版）》的学习，读者不仅能够掌握Netty 基础功能的使用和开发，更能够掌握Netty 核心类库的原理和使用约束，从而在实际工作中更好地使用Netty。

《Netty权威指南（第2版）》

作者简介

李林锋：Netty中国推广者，现华为技术有限公司平台中间件架构与设计部设计师，公司总裁技术创新奖获得者。长期从事高性能通信软件的架构设计和开发工作，有多年在NIO领域的设计、开发和运维经验，精通NIO编程和Netty、Mina等主流NIO框架。目前负责华为软件公司下一代SOA中间件和PaaS平台的架构设计工作。

《Netty权威指南（第2版）》

书籍目录

《Netty权威指南（第2版）》

精彩短评

1、完全不适合对于netty陌生的新手读，先读完netty in action在回头来看这本书，有一些地方可以查漏补缺。缺点就不说了，豆瓣评论中说道的各种缺点仍然适用于第二本。属于在国内科技书籍当中出版质量也比较低的一本书。但是还是有参考意义。。给个两分意思一下。
2、四个字：华而不实，浪费钱和时间
3、netty的书国内比较少，有这么一本入门的不错了，叫权威指南书名太大了，作者用心，但是内容还不够达到这个级别。
4、是的作者抄代码凑字数太多。阅读顺序建议按章节从后往前读，后面的源码分析和架构的图解还比较好，前面例子随便瞄一下就好。
5、有助于了解和熟悉Netty
6、作为入门还不错，整理的部分内容清晰易懂。不过标题起的太大了。。
7、这本书可以不用买了，抛开书的质量不谈，这版Netty5本身已经被放弃了
https://github.com/netty/netty/issues/4466
8、东西越写越少，前面没啥深度的东西占了大量篇幅，后面需要详细分析的东西又一笔带过，浮于表面，没有什么深入的地方，很多东西都是片段，没有一个主线能串起来的，浮躁
9、不多的入门书，要想了解netty还得看源码
10、内容没说的那么差，代码是贴的有点多了
11、书很一般，里面讲的东西其实再官网文档里都找得到。
前面几章都在介绍Java bio/nio/aio（百度一下一大把），中间介绍了下netty基础（官网文档都能找到的东西），然后就是各种代码例子（其实去下载netty源码包，example下面都有这些类似的例子），最后几章在介绍netty的源码（还没看，不知道怎么样）。
再说说里面的例子，就更烂了，跟大学写作业一样，随便写写的，没有一点参考价值。
12、nio的代表，越来越多的cs 采用netty 交互，值得一读。
13、源码部分还可以，提供了思路，不过其他的部分，只是浅浅而谈，介绍api，而且贴代码和console太多，影响排版
14、看的1版电子版，买了纸质的2版。。。或许换个名字，评分会好些？
15、回过头又重新看了一遍netty权威指南，书确实是只有一般性，主要是这方面的书籍也不多，书中的一些类机构图不清晰。线程模型讲解的非常混乱。可以作为入门书籍，单靠这本书很难理解，结合netty in action这两本书互补可以加深理解，再去看源码。
16、读过即便，市面上几乎是唯一一本全面介绍Netty功能的书。作者还分析了一部分Netty源码，可见功力扎实。
17、此书虽然评价不是很高，但是毋庸置疑的是，它是一本国内讲netty比较全面的技术书，可以说，对入门帮助非常大。
18、实在没别的选择就读读吧
19、方法浅显易懂 多是API的一些介绍 作为入门还是不错的
20、很好的工具书，netty网络上的资源比较少，这本书上有丰富的例子，入门很不错。
21、只是大体翻了一下，没有细读。整体上来讲我觉得还是不错的，不像一些评论那么不友好。问题是章节安排详略不当，比如编解码部分，必要性不大，原理讲清楚了一个案例就可以了。还有源码分析部分，近三分之一篇幅，书太厚了。我喜欢In Action系列的风格，要点清楚，不扣细节，注重实战。细节的东西有需要的读者自己会去补充。作为技术同行，深知做技术的辛苦，能系统地写一本技术书，必定踩过很多坑，熬过很多不眠夜。支持原创。
22、很多内容需要仔细深入阅读和理解才能有效果

精彩书评

1、ByteBuffer.allocate(1024)是分配了1MB缓冲区，这是怎么算的？难道不是1KB吗？这么明显的错误也能写出来~~~~~~

2、书很一般，里面讲的东西其实再官网文档里都找得到。 前面几章都在介绍Java bio/nio/aio（百度一下一大把），中间介绍了下netty基础（官网文档都能找到的东西），然后就是各种代码例子（其实去下载netty源码包，example下面都有这些类似的例子），最后几章在介绍netty的远吗（还没看，不知道怎么样）。 再说说里面的例子，就更烂了，跟大学写作业一样，随便写写的，没有一点参考价值

3、”《Netty 权威指南（第2 版）》是异步非阻塞通信领域的经典之作“，这个Title是谁给你封的？大段大段的代码。一个功能的代码，本来就占两页，说是为了改进效率在某个函数就增加了一行代码，就又重新打印了两页代码，并且控制台的输出结果愣是占了三四页。你才是大神。恩，用了十几页口水说的话，根本就没营养好伐？幸好我没买你的书，第一次差评，因为这本书已经影响到国产技术书的信誉了。

4、虽然书里有一些api罗列、源码分析一些不讨喜的部分，但是整体还是很好的，适合于对Netty有初步的使用能力、希望系统化了解学习的朋友们。也许给一星的都是大牛吧，这种级别对他们太low了，不过大牛直接看看英文文档，扫扫源码不就好了吗，不需要再看国人写的书了吧。

《Netty权威指南（第2版）》

章节试读

1、《Netty权威指南（第2版）》的笔记-第6页

目前支持I/O多路复用的系统调用有select、pselect、poll、epoll，在Linux网络编程过程中，很长一段时间都使用select做轮询和网络事件通知，然而select的一些固有缺陷导致了它的应用受到了很大的限制，最终Linux不得不在新的内核版本中寻找select的替代方案，最终选择了epoll。

2、《Netty权威指南（第2版）》的笔记-第115页

Facebook的Thrift介绍
对于当时的Facebook来说，创造Thrift是为了解决Facebook各系统间大数据量的传输通信以及系统之间语言环境不同需要跨平台的特性，因此Thrift可以支持多种程序语言，如C++、C#、Cocoa、Erlang、Haskell、Java、Ocami、Perl、PHP、Python、Ruby和Smalltalk。

3、《Netty权威指南（第2版）》的笔记-第27页

探索多路复用器Selector，它是Java NIO编程的基础，熟练地掌握Selector对于NIO编程至关重要。多路复用器提供选择已就绪的任务的能力。简单来讲，Selector会不断地轮询注册在其上的Channel，如果某个Channel上面发生读或者写事件，这个Channel就处于就绪状态，会被Selector轮询出来，然后通过SelectionKey可以获取就绪Channel的集合，进行后续的I/O操作。

4、《Netty权威指南（第2版）》的笔记-第110页

我们评判一个编解码框架的优劣时，往往会考虑以下几个因素：
1)是否支持跨语言，支持语言种类是否丰富。
2）编码后的码流大小。
3）编解码的性能。
4）类库是否小巧，API使用是否方便。
5）使用者需要手工开发的工作量和难度。

5、《Netty权威指南（第2版）》的笔记-第45页

使用NIO编程的优点总结如下：
1、客户端发起的连接操作是异步的，可以通过在多路复用器注册OP_CONNECT等待后续结果，不需要像之前的客户端那样被同步阻塞。
2、SocketChannel的读写操作都是异步的。如果没有可读写的数据它不会同步等待，直接返回，这样I/O通信线程就可以处理其他的链路，不需要同步等待这个链路可用。
3、线程模型的优化：由于JDK的Selector在Linux等主流操作系统上通过epoll实现，它没有连接句柄数的限制（只受限于操作系统的最大句柄数或者对单个进程的句柄限制），这意味着一个Selector线程可以同时处理成千上万客户端连接，而且性能不会随着客户端的增加而线性下降。因此，它非常适合做高性能，高负载的网络服务器。

6、《Netty权威指南（第2版）》的笔记-第79页

4.1 TCP粘包/拆包
TCP是个“流”协议，所谓流，就是没有界限的一串数据。大家可以想想河里的流水，它们是连成一片的，其间并没有分界线。TCP底层并不了解上层业务数据的具体含义，它会根据TCP缓冲区的实际情况进行包的划分，所以在业务上认为，一个完整的包可能会被TCP拆分成多个包进行发送，也有可

能把多个小的包封装成一个大的数据包发送，这就是所谓的TCP粘包和拆包问题。

7、《Netty权威指南（第2版）》的笔记-第7页

熟悉TCP编程的读者可能都知道，无论是服务端还是客户端，当我们读取或者发送消息的时候，都需要考虑TCP底层的粘包/拆包机制。

8、《Netty权威指南（第2版）》的笔记-第4页

阻塞IO模型
非阻塞IO模型
复用IO模型
信号驱动IO模型
异步IO

9、《Netty权威指南（第2版）》的笔记-第114页

Protobuf另一个比较吸引人的地方就是它的数据描述文件和代码生成机制，利用数据描述文件对数据结构进行说明的优点如下：
文本化的数据结构描述语言，可以实现语言和平台无关，特别适合异构系统间的集成。
通过标识字段的顺序，可以实现协议的前向兼容。
自动代码生成，不需要手工编写同样数据结构的C++和Java版本。
方便后续的管理和维护。相比于代码，结构化的文档更容易管理和维护。

10、《Netty权威指南（第2版）》的笔记-第107页

6.1 Java序列化的缺点
在远程服务调用（RPC）时，很少直接使用Java序列化进行消息的编解码和传输：
1、无法跨语言。
无法跨语言，是Java序列化最致命的问题。对于跨进程的服务调用，服务提供者可以能会使用C++或者其他语言开发，当我们需要和异构语言进程交互时，Java序列化就难以胜任了。
由于Java序列化技术是Java语言内部的私有协议，其他语言并不支持，对于用户来说它完全是黑盒。对于Java序列化后的字节数组，别的语言无法进行反序列化，这就严重阻碍了它的应用。
目前几乎所有流行的Java RPC通信框架，都没有使用Java序列化作为编解码框架，原因就是在于它无法跨语言，而这些RPC框架往往需要支持跨语言调用。

11、《Netty权威指南（第2版）》的笔记-第2页

Java的NIO编程并没有流行起来，究其原因如下。
1、大多数高性能服务器，被C和C++语言盘踞，由于它们可以直接使用操作系统的异步I/O能力，所以对JDK的NIO并不关心。
2、移动互联网尚未兴起，基于Java的大规模分布式系统极少，很多中小型应用服务对于异步I/O的诉求不是很强烈。
3、高性能、高可靠性领域，例如银行、证券、电信等，依然以C++为主导，Java充当打杂的角色，NIO暂时没有用武之地。
4、当时主流的J2EE服务器，几乎全都基于同步阻塞I/O构建，例如Servlet、Tomcat等，由于它们应用广泛，如果这些容器不支持NIO，用户很难具备独立构建异步协议栈的能力。
5、异步NIO编程门槛比较高，开发和维护一款基于NIO的协议栈对很多中小型公司来说像是一场噩梦

。
6、业界NIO框架不成熟，很难商用。
7、国内研发界对NIO的陌生和认识不足，没有充分重视。

如果说个人能够改变自己命运的话，对于程序员来说，唯有通过不断地学习和实践，努力提升自己的技能，才有可能找到更好的机会，充分发挥和体现自己的价值。

12、《Netty权威指南（第2版）》的笔记-第2页

根据UNIX网络编程对I/O模型的分类，UNIX提供了5种I/O模型，分别如下。
1、阻塞I/O模型：最常用的I/O模型就是阻塞I/O模型，缺省情况下，所有文件操作都是阻塞的。我们以套接字接口为例来讲解此模型：在进程空间中调用recvfrom，其系统调用直到数据包到达且被复制到应用进程的缓冲区或者发生错误时才返回，在此期间一直会等待，进程在从调用recvfrom开始到它返回的整段时间内都是被阻塞的，因此被称为阻塞I/O模型。
2、非阻塞I/O模型：recvfrom从应用层到内核的时候，如果该缓冲区没有数据的话，就直接返回一个EWOULDBLOCK错误，一般都对非阻塞I/O模型进行轮询检查这个状态，看内核是不是有数据到来。
3、I/O复用模型：Linux提供select/poll，进程通过将一个或多个fd传递给select或poll系统调用，阻塞在select操作上，这样select/poll可以帮我们侦测多个fd是否处于就绪状态。select/poll是顺序扫描fd是否就绪，而且支持的fd数量有限，因此它的使用受到了一些制约。Linux还提供了一个epoll系统调用，epoll使用基于事件驱动方式代替顺序扫描，因此性能更高。当有fd就绪时，立即回调函数rollback。
4、信号驱动I/O模型：首先开启套接接口信号驱动I/O功能，并通过系统调用sigaction执行一个信号处理函数（此系统调用立即返回，进程继续工作，它是非阻塞的）。当数据准备就绪时，就为该进程生成一个SIGIO信号，通过信号回调通知应用程序调用recvfrom来读取数据，并通知主循环函数处理数据。
5、异步I/O：告知内核启动某个操作，并让内核在整个操作完成后（包括将数据从内核复制到用户自己的缓冲区）通知我们。这种模型与信号驱动模型的主要区别是：信号驱动I/O由内核通知我们何时可以开始一个I/O操作；异步I/O模型由内核通知我们I/O操作何时已经完成。

13、《Netty权威指南（第2版）》的笔记-第1页

异步高性能内部协议栈、异步HTTP、异步SOAP、异步SMPP，所有的协议栈都是异步非阻塞。基于Reactor模型统一调度的长连接和短连接协议栈，无论是性能、可靠性还是可维护性，都可以秒杀传统基于BIO开发的应用服务器和各种协议栈，这种差异本质上是一种代差。

14、《Netty权威指南（第2版）》的笔记-第6页

1. 支持一个进程打开的socket描述符（FD）不受限制（仅限制于操作系统的最大文件句柄数）
1G内存的机器大约支持10万个句柄。
$ cat /proc/sys/fs/file-max 可查系统支持我句柄数

2. IO效率不会随着FD数目的增加而线性下降。

3. 使用mmap加速内核与用户空间的消息传递。
epoll通过内核和用户空间mmap共享同一块内存来实现。

4. epoll的API更简单。

15、《Netty权威指南（第2版）》的笔记-第116页

JBoss Marshalling
优点如下：
1、可拔插的类解析器，提供更加便捷的类加载定制策略，通过一个接口即可实现定制。
2、可拔插的对象替换技术，不需要通过继承的方式。
3、可拔插的预定义类缓存表，可以减少序列化的字节数组长度，提升常用类型的对象序列化性能。
4、无须实现java.io.Serializable接口，即可实现Java序列化。
5、通过缓存技术提升对象的序列哈性能。

《Netty权威指南（第2版）》

版权说明