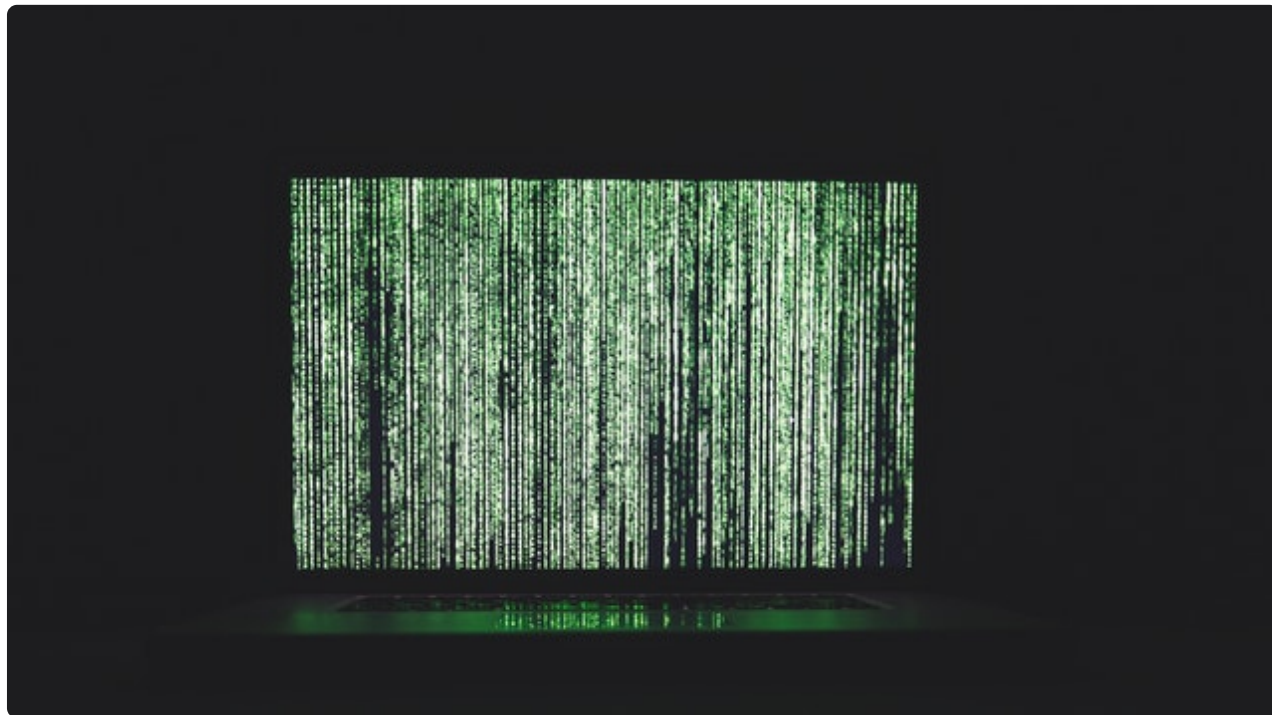


## 02 Netty是如何发迹的

更新时间：2020-07-13 13:46:27



“更多一手资源请+V：AndyqcI  
完成工作的方法，是爱惜每一分钟。——达尔文  
aa：3118617541”

### 前言

你好，我是彤哥。

话说天下大势，分久必合，合久必分：网络协议早期便衍生出了 OSI 七层协议，后归于 TCP/IP 协议（最火）；IO 模型早期便衍生出了五种 IO 模型，后归于 NIO 模型（最热）；自 NIO 模型诞生以后，各种基于 NIO 的网络框架群雄并起、割据分争，最后归于 Netty（最闪）。

### Java NIO 时代

溥天之下，莫非王土；网络编程，莫非 Java NIO！

2002 年，注定是一个伟大的年代，JDK1.4 版本在这年发布，并加入了对于 NIO 的支持，它虽然不算很强大的框架，但是早期在没有框架可用的情况下也必须使用它来进行网络编程，即使偶然还搞出个空轮询的 bug，这就是属于 Java NIO 的时代 —— 难用也必须得用。

NIO 模型中的 **N** 一般指代 **Non-blocking**，即非阻塞的意思，NIO 即非阻塞 IO 的意思。

广义上来说，NIO 包括 Non-blocking IO（非阻塞 IO）、IO Multiplexing（IO 多路复用）、Signal-Driven IO（信号驱动 IO）。

狭义上来说，NIO 仅指 Non-blocking IO。

Java NIO 中的 **N** 是指 **New**，即新的意思，Java NIO 即新的 IO，是相对于 OIO (Old IO) 或者 BIO (Blocking IO) 来说的，在 Java 的实现中是基于 IO Multiplexing 模型实现的。

说起 Java 对于 NIO 的支持，我们有必要揭起那道伤疤，看看使用 Java NIO 编程曾经给大家带来了“多么痛的领悟”：

- API 复杂难用，尤其是 Buffer 的指针切来切去的，反人类设计
- 需要掌握丰富的知识，比如多线程和网络编程
- 可靠性无法保证，断线重连、半包粘包、网络拥塞统统需要自己考虑
- 空轮询 bug，CPU 又 100% 了，一直未根除此问题

所以，在 Java NIO 末期进入了群雄并起的时代，各种框架层出不穷，都想在 NIO 的热潮中分一杯羹。

## 群雄并起

### Netty2

2004 年，一个棒子国的游戏浪子 Trustin Lee，怀揣着梦想，站在了 BIO 和 NIO 的十字路口，左顾右盼，最终走向了 NIO。那年，他带着自己最闪亮的作品 **Netty2** 敲响了棒子国最大的移动通信商 ——Arreo 通信公司 —— 的大门，毫无疑问，他被成功录取了。

彼时，Netty2 号称是 Java 社区中第一个基于事件驱动的网络应用框架，它的梦想是做夜空中最闪亮的一颗星：

```
<dependency>
<groupId>net.gleamynode</groupId>
<artifactId>netty2</artifactId>
<version>1.9.2</version>
</dependency>
```

更多一手资源请+V：Andyqc1  
qq：3118617541

gleamy node，夜空中最闪亮的那一颗。

启示，梦想还是要有的，万一实现了呢。

### Grizzly

同样是 2004 年，一款叫作玻璃鱼（Sun 公司开发的 GlassFish）的应用服务器诞生了，它是为了打败 Tomcat 而生的。使用了最新的 NIO 技术，并封装成了完整的框架 ——Grizzly，专门解决编写成千上万用户访问服务器时候产生的各种问题。

Grizzly 使用 Java NIO 作为基础，提供了高性能的 API，并隐藏了编程的复杂性，使得它一出世就很火。比如，我们熟知的 Jetty、Comet 等都是基于 Grizzly 构建的。

但是，光顾着沉浸在欢乐喜悦之中，却忘了三件非常重要的事：

- 文档少，几乎没什么可用的文档
- 更新慢，好几个月才一次提交
- 用户少，社区没做好，用户不多

基于以上三点原因，使得它被 **Netty** 甩了一条街，不管是社区活跃度还是流行度：

### Netty VS Grizzly

[netty.io](#)[Source Code](#)[Changelog](#)

Framework for building high performance network applications.

[grizzly.java.net](#)[Source Code](#)[Changelog](#)

NIO framework. Used as a network layer in Glassfish.

Compare Netty and Grizzly's popularity and activity

Popularity ★ 9.8	Stable ▶	Popularity ★ 2.6	Stable ▶
Activity ♥ 9.4	Stable ▶	Activity ♥ 8.6	Stable ▶

Netty	Grizzly
Repository	
20,168	★ Stars 98
1,751	👁 Watchers 15
9,232	🔗 Forks 29
5 months ago	🕒 Last Commit over 2 years ago
More	
L2	Code Quality L4
Java	</> Language Java
Networking	🏷 Tags Networking

更多一手资源请+V：Andyqc1  
aa：3118617541

启示，东西好，还要用心经营。

## Tomcat

2005 年前后，**BIO** 时代的霸者 ——**Tomcat**，意识到了 **Grizzly** 带来的危机，逆势发布了 **6.0** 版本，它也开始支持 **NIO** 了，同样使得它的性能有了质的飞跃。

但是，**Tomcat** 的 **NIO** 通信层并没有从它本身的代码中解耦出来，形成了一种“老奶奶裹脚，又臭又长”的代码，这也使得它在 **NIO** 的浪潮中只能保住自己的一亩三分地，并不具有进攻他人的属性。

启示，代码解耦真的很重要。

## MINA

2005 年，另一个年轻人 **Alex**，他当时正在为 **Apache Directory** 开发网络框架，但是他自己怎么写都感觉不好，一次偶然的机会，它瞅见了 **Netty2**，被它的闪亮所折服，就找到了当时还在 **Arreo** 通信公司的 **Trustin Lee**，邀请他一起来开发网络框架，后面就诞生了一个伟大的网络通信框架 ——**MINA**，并把它贡献给了 **Apache**。

**MINA** 可帮助用户轻松开发高性能和高可伸缩性的网络应用程序，支持各种传输协议，比如 **TCP** 和 **UDP** 等。

这可能是 Trustin Lee 一生的痛，所以后来他一再强调 MINA 没有 Netty 好用：

136

MINA has more out-of-the-box features at the cost of complexity and relatively poor performance. Some of those features were integrated into the core too tightly to be removed even if they are not needed by a user. In Netty, I tried to address such design issues while retaining the known strengths of MINA.

Currently, most features available in MINA are also available in Netty. In my opinion, Netty has cleaner and more documented API since Netty is the result of trying to rebuild MINA from scratch and address the known issues. If you find that an essential feature is missing, please feel free to post your suggestion to the forum. I'd be glad to address your concern.

It is also important to note that Netty has faster development cycle. Simply, check out the release date of the recent releases. Also, you should consider that MINA team will proceed to a major rewrite, MINA 3, which means they will break the API compatibility completely.

share improve this answer

answered Jan 4 '10 at 15:14



trustin

10.8k ● 6 ● 33 ● 48

大致意思就是说：

- MINA 的功能繁杂，复杂性高，性能差
- MINA 的功能过于耦合
- Netty 的 API 更整洁、更文档化
- Netty 的更新周期短
- MINA3 会破坏 API 的兼容性

说白了，MINA 虽然是 Trustin Lee 主导开发的，但是它已经贡献给了 Apache，自己不再参与其中了，而且还是自己的亲儿子 Netty 的强力对手，不抨击它抨击谁。

启示，千万不要轻易给自己制造竟然对手。

## Netty3

2008 年，Trustin Lee 裹挟着迷茫和彷徨加入了 JBoss，并在同年发布了 Netty3。

这是一个全新的 Netty，背靠大厂 JBoss，并借鉴了 MINA 中很多优秀的设计，使得它一发布就引起了社会各界人士的关注，更新频繁，社区也异常活跃，占领了网络通信的半壁江山。

那时，Netty 只想安安稳稳地待在 JBoss 这个大厂中，稳定至上：

```
<dependency>
  <groupId>org.jboss.netty</groupId>
  <artifactId>netty</artifactId>
  <version>3.0.0.Final</version>
</dependency>
```

启示，背景很重要，你懂得。

## Netty4

2012 年，内心中的那份不安和执念终于爆发，Trustin Lee 决定了要想随心所欲地做自己想做的事，还是得出来单干。

那年，他带着 Netty4 又一次出现在了大家的面前，相对于 Netty3，他做了很多改进，比如线程模型、池化的 Buffer 等，性能提升到了极致。

此时，Netty 才算彻底独立，当然，它也终于成为了那颗最闪亮的星：

```
<dependency>
  <groupId>io.netty</groupId>
  <artifactId>netty-all</artifactId>
  <version>4.0.0.Final</version>
</dependency>
```

启示，追寻自己的内心，随心所欲，为所欲为！

## Netty5

2013 年，出来单干的 Trustin Lee 是有野心的，很多 JDK 的新特性他也想去尝试，比如 ForkJoinPool，所以这一年有了 Netty5。

但是，好景并不长，在 Netty5 发布了两个 Alpha 版本，被一个叫 Norman Maurer（《Netty in Action》的作者）的家伙在 2015 年双 11 这天提议给废弃了，他的理由很简单也很任性：



normanmaurer commented on 11 Nov 2015

Member + 😊 ...

After talking with @Scottmitch and also with @nmittler I would like to propose "dropping" current master and so not release any new 5.0 version for now.

The major change of using a ForkJoinPool increases complexity and has not demonstrated a clear performance benefit. Also keeping all the branches in sync is quite some work without a real need for it as there is nothin in current master which I think justifies a new major release.

Things that we should investigate to prepare for this change:

- Deprecate `exceptionCaught` in `ChannelHandler`, only expose it in `ChannelInboundHandler`
- Expose `EventExecutorChooser` from `MultithreadEventExecutorGroup` to allow the user more flexibility to choose next `EventLoop`
- Add another method to be able to send user events both ways in the pipeline. (#4378)

Please chime in here and let me / us hear any concerns.

👍 23 🙄 10 👁 1

- 使用 ForkJoinPool 提升了复杂性
- 没有带来明显的性能提升
- 同时维护太多分支太耗费精力

启示 1，做自己能力范围内的事。

启示 2，千万不要使用 Alpha、Beta 等非稳定版本。

## Netty 时代

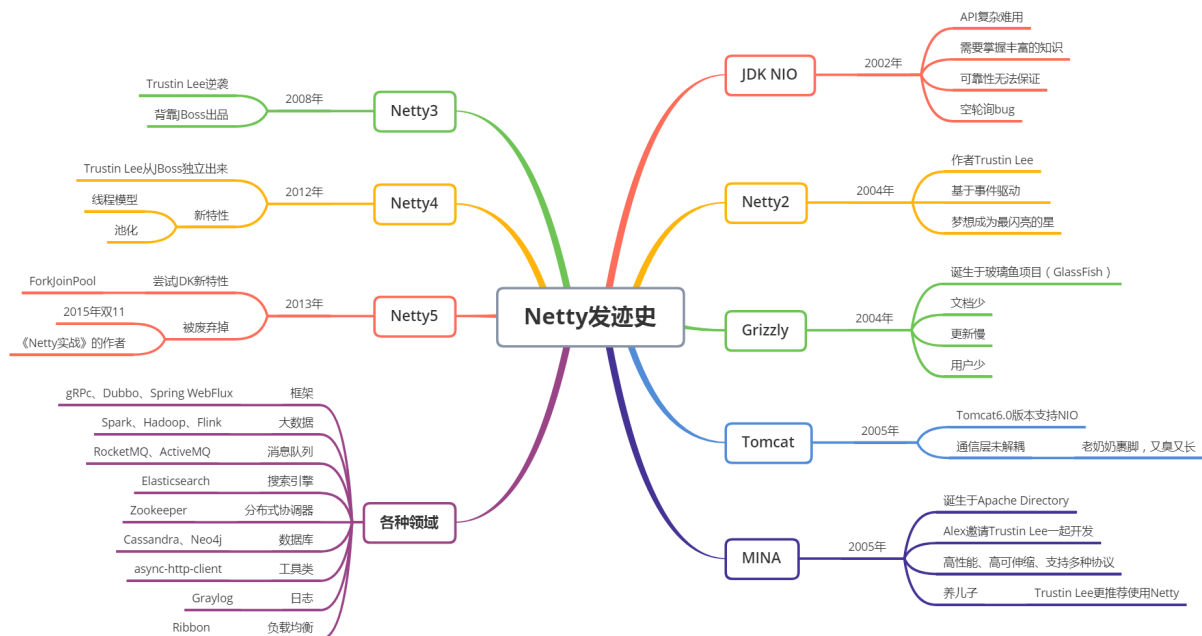
现在这个时代是属于 Netty 的，大家越来越喜欢 Netty，以前使用 MINA 等其它通信框架的也在逐步转换为 Netty，说 Netty 统一了 Java 领域中的网络通信一点也不为过，在很多领域都能见到它的身影：

- 框架，gRPC、Dubbo、Spring WebFlux、Spring Cloud Gateway
- 大数据，Spark、Hadoop、Flink
- 消息队列，RocketMQ、ActiveMQ
- 搜索引擎，Elasticsearch
- 分布式协调器，Zookeeper
- 数据库，Cassandra、Neo4j
- 工具类，async-http-client
- 日志，Graylog
- 负载均衡，Ribbon

## 后记

Netty 从最初梦想着成为最闪亮的星，经历了群雄割据奋战，现在已经被广大框架所运用，它无疑已经成为 Java 网络通信领域的事实标准，所以我们有必要学好 Netty，并熟练掌握如何在实战中使用 Netty。

思维导图 **更多一手资源请+V：Andyqc1aa：3118617541**



}

更多一手资源请+V : AndyqcI  
aa : 3118617541