初中级程序员 BAT 面试复盘

1. 简介

本 Chat 作者普通院校毕业，没有光鲜亮丽的职业背景，凭着自己的激情和两个月的准备最终拿到京东和阿里巴巴 Offer。下面就是作者此次跳槽经历所有准备工作和心得，希望对你有帮助，作者工作经验有限，此文适合少于5年工作经验者。初次写作如有不对之处，欢迎批评指正。

本文是作者复盘自己两个月的面试经历，希望通过我的分享，对大家有帮助，祝大家跳槽成功。

> 文末有彩蛋。

2. 克服心理是投简历的第一步

第一章是枯燥的心理建设，如果你内心足够强大，直接略过看下一章。

2.1 摆脱舒适区

笔者妄自揣度读者的心里，公司现在经营不景气、公司没有发展、公司可能会倒闭、自己的职业生涯没有规划、公司现在做的东西和我的理想不一致等各种原因，但却说服不了处于舒适区的自己走出来。

大多数的我们都是呆在舒适区，在这个区域压力和变化都很小，让我们的生活很平静，同时改变相对就少很多了。在舒适区待久的我们真的很难迈出一步去挑战自己，因为我们需要再次去学习，去挑战自己。

离开舒适区就意味着较高的风险和焦虑，它可以导致积极的和消极的结果，然而学习区是真正的“产入”阶段。是让人精力和行为达到最佳状态的区域，在学习区你会消失因为期待带来的不安，也不会有每天无所事事，自怨自艾的“工作迷茫”，更不会每天晚上感叹一天又是无所事事。

当你真正挑战自己时，你所做出的成就会让人惊叹。正如一句话：“你不逼自己，永远不知道自己有多强。然而要把握好自己的容忍度，如果到了恐慌区不仅工作效率不高，还会失去信心。所以现在的你应该考虑自己是不是那个在舒适区的自己，如果你还想挑战自己，而不是每天自怨自艾，请行动起来。笔者也是特别不愿意投第一封简历，也不想去面试，有一天实在忍不下去了，咬着牙投出去第一封，接下来就会投 2，3，4 …

> 结语：你不逼自己永远不知道自己有多强，如果你不想整天自怨自艾，那么就让自己行动起来，迈出一步，后面就好走多了。

2.2 地球没有你一样可以转

当然此段落可能说的不是你，之前和好几个朋友聊过，他们都觉得公司对自己不错，同事关系很好，而且公司业绩正在发展，现在不能没有我，迟迟不忍心动手。那么如果你有这种想法你就错了，地球没有谁都在转动，如果你真的觉得自己没有在成长，那么果断的离开，对于你和公司都是一件好事。

对于公司，没有那么忠诚的你他们可以找更好的，而且你要知道，一个产品的灵魂其实更多的是 PM 赋予的，我们这种做程序的是这个行业最容易替换的职位，所以公司不会因为一个技术的离开会一蹶不振；对于你，如果你没有更好的平台、经历来武装自己，时间久了就会和正在成长的公司、朋友们脱节，现在也许你们在一个起跑线，过一段时间，再好的关系也会变得陌路。

> 结语：地球没有谁都在转动，换工作没有对不起谁，对不起的只是自己，只有自己才能为自己买单。

2.3 明日复明日，明日何其多

之前遇到一堆问题无从下手的时候，时不时用“明日复明日，明日何其多”给自己打趣。其实你原本计划的事情没有完成，究源还是你的时间管理不规范导致，随便拿一个“奇妙清单”就过来使用其实效果很不好，如果你觉得自己的计划和时间也安排的乱糟糟，请看看这本书。《搞定 I :无压工作的艺术》，也可以点击查看笔者收集的 PDF。这本书是时间管理领域最具影响力的著作之一，书中介绍的 GTD（Getting Things Done）时间管理方法也成为全球千万读者用来轻松高效完成工作与个人事务的最佳工具。

> 结语：所以既然决定的时候就一定要付诸行动，如果因为繁杂的事情没有安排好，那么就尝试使用 GTD 合理的规划一下自己。你要知道，我们都有拖延症，你永远都不会准备好，只有迈出第一步才能开始准备。

2. 4 盲目自信也是在耽误你

笔者起初也是颇有自信，觉得自己工作三年有余就独立的负责很大的模块，并且是公司的核心开发人员，一直在给自己心理暗示：“我现在就是不想找工作，现在我在创业公司能够独当一面，啥时候想找不是很轻松的事情？”。

然而就是这个想法让我从有想法换工作到开始动身足有小一年。等到动身的时候我才知道自己曾经的想法是错误的。

“ 格局的局限是你致命错误 ”。我开始试水是在拉钩上面投了 10 几封简历，结果一周过去毫无音讯。

我开始反思问题，于是托了一位工作很久的朋友帮忙看一下简历，反馈竟是这样：“ 你的简历写的太乱了，没有条理，苍蝇蚊子一把抓，而且你的经历太散，没有重点，你得好好整理下自己的项目经历和技术能力。”。

于是，就有了下面一章《 整理简历，好的开始是成功的一半》，怎么整理简历和自己的技能点。

> 结语：盲目的自信也是你一拖再拖的理由，是骡子是马，拉出来溜溜一下子就知道了。

3. 整理简历，好的开始是成功的一半

这是我的第一份简历，打个样你就知道有多差了，其实当时我自己觉得内容很多，技术点很多，还说明了做什么了，并且最后还言明自己在这个项目中的厉害和学到了什么，不应该是很不错吗？

服务端工程师 2016.02-至今

DailyCast 是全球首个个性化的移动短视频杂志，包含海外短视频聚合，推荐和分发等主要功能。APP 主要通过聚合 FB, Youtube, Vimeo 等平台的视频，进一步加工，推荐和分发。主要涉及到的功能有，为客户端提供高可用 API，消息中间件，邮件系统，推送系统，离线抓取系统。框架使用 SpringMVC 和 MyBatis，数据库使用 MySQL, DynamoDB，同时使用 ElasticSearch 为搜索提供服务，使用 Redis 作为缓存数据库。服务器搭建在 AWS，所以相应的工具也使用 AWS 的工具，比如 Route53, S3, EC2, LB, PIPE 等。在项目中学习到从响应时间和减少带宽方面优化接口，增加消息中间件以提供的高可用 API 保证；精细化运营邮件系统，提高送达率和增加送达统计；推送系统细粒度区分用户，减少服务器开销同时增加送达率；深度面向对象的设计等除了本身技术实现以外的技术，思想从技术的实现开始向性能和设计转变。

然而事情并不是我想的那样，很大一个篇幅描述，杂糅到一起既显得杂乱无章，也让阅读者抓不住重点，其实你可以把简历比喻成代码，这样一坨还不重构怎么让人维护？ 不喷你才怪。

首先需要整理版面，起码让阅读者看着舒服，笔者找了很多方案，觉得下面两种方案不错：

1. 使用表格很规整的 Word 文档，清晰呈现自己的信息和工作经历，具体模板 点击获取。

2. 使用 Markdown 编写简历，原本我们就非常熟悉 Markdown 的语法，而且他的样式对于我们阅读来说已经很可以了，并且层次目录清晰。

其次是个人基本信息，个人基本信息一定要字字珠玑，不要写没用的，比如驾照，血型。罗列重点信息即可，并且学历一定写的清清楚楚，不要让面试官去猜疑。如果有不

错的社交属性可以放上，比如不错的 Github 或者是简书。

然后就是最重要的一点清晰的描述自己的工作内容并突出重点，一定要有条理并且分层次，最简单的方式就是公司用 `#`，项目名称用 `##`，技能，内容，描述分别用 `###`，这样结构就非常清晰，然后使用 FAB 或者是 STAR 原则，描述自己的贡献，这样就清晰多了。

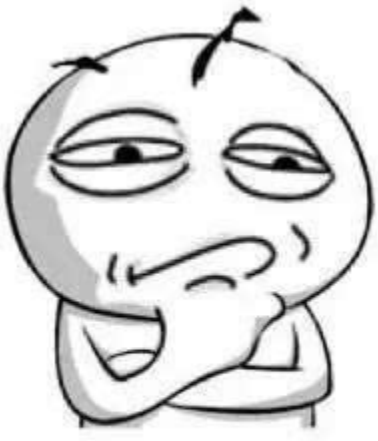
打个样，看一下笔者修改以后是不是好了一些：

项目描述:
该项目主要是聚合 `Facebook`，`Youtube`，`Vimeo` 等平台海量视频做推荐和分发。
实现技术:
AWS + SpringMVC + RabbitMQ + MyBatis + Dubbo + Redis + MySQL + DynamoDB + ElasticSearch
职责描述:

1. 作为三位核心开发之一，独立负责大部分接口的设计、开发和联调工作。
2. 独立重构数据抓取中心，完成其设计和实现，不仅保证了业务的独立性同时提高了抓取效率近3倍。
3. 为项目添加消息中间件，使得 `QPS` 提高近1倍。
4. 独立完成 `Push` 系统，定时、高效、精准的推送消息。
5. 独立负责邮件系统的调研和开发工作，并为其添加统计功能，最终把邮件送达率优化到80%以上。
6. 优化 `cms` 系统，增加长文章离线保存、恢复功能，大大减少了运营人员文章丢失情况，从而避免了因系统功能导致运营人员重复工作的发生。
7. 为项目构建自动化部署系统，节约了开发人员的部署成本。
8. 负责安排和指导实习生工作，使其高效地产出和自我成长。

切记不能有错别字，这是写简历的大忌。

最后是技能一定要突出重点，如果你找的是 JAVA 工作，其实一线互联网公司是不会关心你对 HTML 有多么熟悉，所以重心放在 JAVA 上面，比如 NIO、多线程、JVM 等。



> 笔者注：当然如果你选择写上去的东西，必须可以自圆其说，必须真实，不然就适得其反了。

简历确实是非常重要的，作者找到了一篇还不错的文章，写的非常详细，里面还有技巧和模板，比笔者总结的好那么一丢丢[坏笑]，如何写好技术简历 —— 实例、模板及工具，可以参考一下。

> 结语：好的开始是成功的一半，一定要在自己的简历上面下一番功夫。

4. 打造自己的社交简历

一份 PDF 简历是求职必须的，然而有一份“社交简历”更能让你锦上添花，可以简单的从“订阅号”说起，为什么订阅号放在社交简历的第一位呢？

4.1 订阅号

每天地铁里面上下班你用什么来打发时间？大多数人都在看订阅号吧，微信订阅号这么大的体量，又源于社交，那不是最好的打造自己社交的一个工具？

笔者了解，85% 以上的订阅号作者都是自己乐于分享，乐于传播，乐于交友的，那么笔者问你，你订阅了那么多自己觉得还不错的微信号，加了几个订阅号主的微信？

> 惊讶脸？还有这操作，如果没有意外，90% 的订阅号主都会把自己的微信获取方式放到订阅号的菜单里面。

为什么笔者建议你添加订阅号主，因为互联网圈子就这么小，人脉是非常重要的一个财富，工作越久你就会越觉得人脉很重要，既然你没有父辈带来的人脉，那么自己扩展也是一个不错的选择。另一个原因，有一句古话说的好：“近朱者赤近墨者黑”，如果你想进大厂，是不是先了解一下大厂的人怎么思考，怎么做事，是一个什么水平？

那么订阅号不少都源于大厂，这是你最方便可以触及大厂人的生活的方式，添加他的微信则是更一步的方式。没事多阅读下他的文章，点赞，互动一下，慢慢就会变成你的关系网。

> 打个样，比如笔者的收集，阿里巴巴的 并发编程网，京东的 开涛的博客，ThoughtWorks 的 诺普博客，还有笔者的微信 码匠笔记，不过笔者还是一个初级选手。 如果你留心，订阅号会成为你最好的人脉。

4.2 写博客

然后是博客，笔者私以为一个做技术的没有一个博客是万万不能够的。一个好的博客不仅体现了你对技术的那份热情，还有你水平的最好的体现。短短的几个小时面试是不能够深入的了解一个人的，但是你的博客是你阅历的写照，有心的面试官会仔细审度你的

博客，挖掘你的亮点。同时你的博客也能提高你的知名度，比如“阮一峰老师”或者是“张鑫旭老师”去找工作，是不是会轻松很多？

当然我们不会把博客打造成他们那个样子，那么权威，但是提高自己，整理思路还是足够的。

> 虽然下文中《怎么复习》章节，笔者说看别人的文章复习是不足的，但是你写博客是把你的知识表达出来，你看明白和你能讲明白是两码事。所以《收藏 != 拥有》章节讲，你需要把你收藏的东西整理成自己的文章，然后发表出来，这样自己掌握的便会更加深入。

4.3 单身交友社区 Github

Github，都说 Github 是单身交友俱乐部，不过也有妹子哦。

作为技术人员，Github 是最好的社区了，你可以在上面看到非常不错的开源，同时也能和作者互动，当然互动的方式很多，比如 Pull Request，Issues，Star，Fork 等等，如果你自己热爱开源，有小想法，可以自己做一个开源的小工具，哪怕只有一个点，也是一份开源的热爱。

笔者的 Github 还算是朴素，只是添加了一些自己业余时间的开源，比如 Chrome 目录插件，Facebook Messenger SDK 封装，同时发送 Markdown 文章到 CSDN、SegmentFault、

简书等平台工具 等，同时有时间也会参与开源社区的翻译，比如MDN、ElasticSearch 官方文档、Spring4All 译文等。

这样你长期坚持下去，会在 Github 结交一些志同道合的人，同时也能增加自己的亮点。何乐而不为？

> 笔者也是初入社会，才疏学浅，如果你有更好的 “ 招式 ” 欢迎留言区留言，让笔者和其他读者能学到更多的 “ 武术 ”。
>
> 结语：PDF 的简历只是你面试的一部分，如果你对这个行业充满着好奇，那么就用业务时间多经营一下自己的社交简历。

5. 你最关心的面试点

笔者在网上看到过 N 多的面经，上来就说面试问到什么，具体到类，对象，有的可能还有答案，但是笔者这个章节并不会 一一罗列面试问题。

> 哎，但是笔者考虑到有的读者想看一下笔者经历的面试题，所以还是把自己的面试问题放在下面《我的面试经历》章节

回到正题，因为笔者认为，面试点虽然很重要，但是它并不代表你掌握了就可以轻松应付，它只是知识面里面的一个点而已。你记住今天的问题，明天面试官换一个问题你，你还是回答不上来，但是大部分面试官提问的知识面其实都差不多。所以如果你想应付自如，就从面试官的知识点中挖掘出知识面，从而达到举一反三，触类旁通。

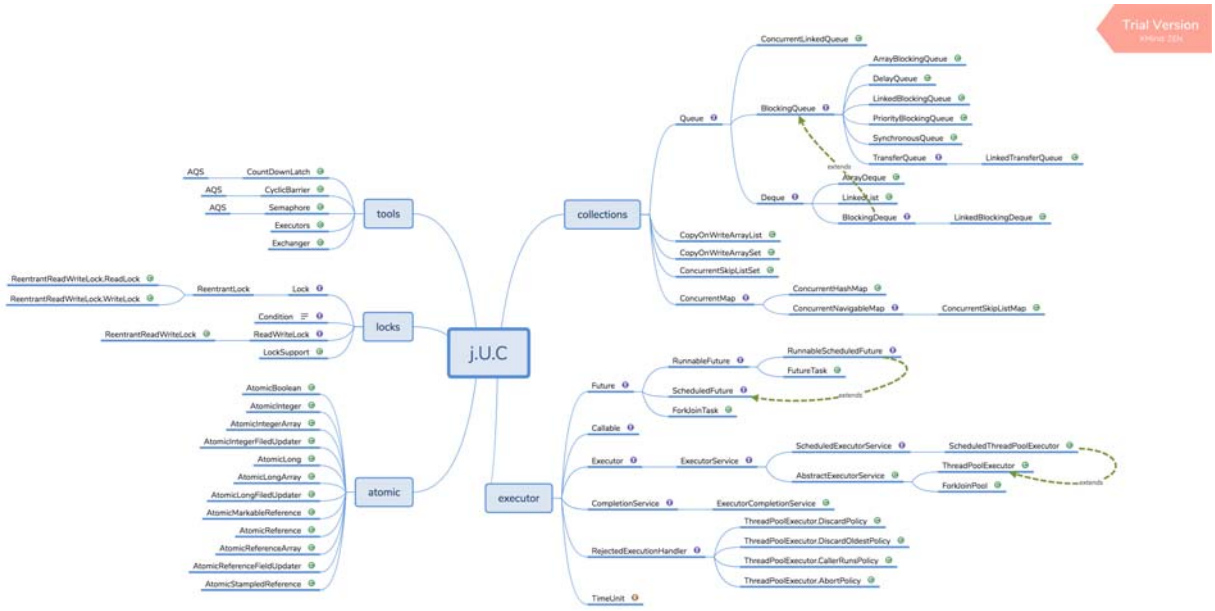
> 你是否有这样的经历：明明每次面试之前都认真研究面经，整理知识点的，但是去面试的时候总是被问住。如果是，那说明你可以根据笔者的方法挖掘一下知识面了。

下面笔者通过一个具体的例子讲解一下怎么从知识点分析出知识面，然后再全面复习。

> 因为不同的领域的知识点也不一样，笔者只是抛砖引玉，具体的技术，还得你根据自己的情况仔细揣摩。

1. Synchronized 和 ReentrantLock 锁机制？
2. HashMap 的原理？当谈到线程不安全时自然引申出 ConcurrentHashMap ， 它的实现原理？
3. 写一下生产者消费者。
4. 介绍一下 BlockingQueue 原理。
5. 介绍一下 PriorityBlockingQueue 原理。
6. 说一下 CountDownLatch 的使用场景和原理。
7. 罗列一些你知道的线程安全的集合类。

上面这些问题都是笔者在面试不同公司的问题，万卷不离其宗，都是在考 JAVA concurrent 包下面的知识点，那么我们需要梳理一下 concurrent 包下面都有什么，都有什么关联的知识点，然后统一复习，即便是面试官再换一些问题也没有关系了。直接用图来说明一切



上图是整个 JAVA concurrent 包下面的类，我们有了这个图就可以各个击破了，比如我们需要看信号量 Semaphore，读写锁 ReentrantReadWriteLock，CountDownLatch 它们的实

现和使用场景，当然他们的实现需要依赖 AQS，CAS，Atomic，具体它们都是什么呢？这就需要你细细去品了。

下文章节《怎么复习》里面提到，切记不能看博客，也别听笔者道听途说，一本书的内容笔者三言两语就讲清楚了，那么也不需要画这么大一个图了，也就没有那么高深了。

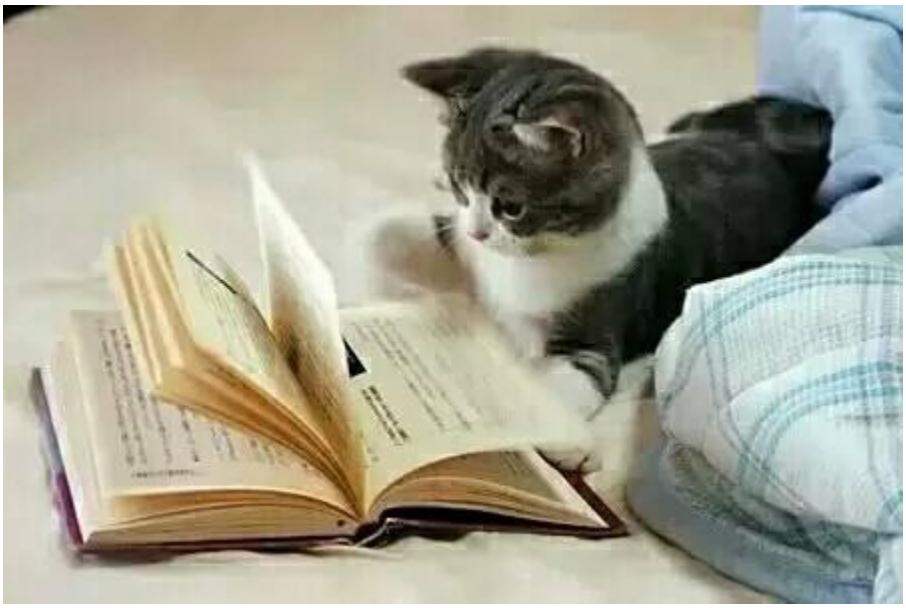
所以请继续往下看《怎么复习》章节怎么做。然后到了 collections 下面的集合类，每一种的实现都离不开 Lock，具体它们之间怎么关联的呢？你一看源码就懂了。最后是 executor，它就是一个工具，但是也离不了上面的 Lock 和集合类，所以只要你仔细的看《Java并发编程的艺术 (Java 核心技术系列)》这本书，想必好多问题你都可以迎刃而解。

再举一个笔者复习 ElasticSearch 的例子，通过《我的面试经历》章节你可以发现，4家公司都问过 ElasticSearch 的问题，比如ElasticSearch存储过程，索引过程，查询过程，原理和数据结构。那么笔者势必要下功夫了，于是笔者从头到尾读了一遍《ElasticSearch 权威指南》。

笔者没有找到合适的书籍，中文文档是 2.0 版本有些过时了，但是基本原理类似，如果英文好的话可以直接看 6.0 最新英文文档。里面会给你讲用法、实现、存储结构、设计原理还有集群搭建、分片原理，反正你可以学到好多东西，这样你就可以参照《学以致用》章节，掌握以后马上考虑哪里可以用到当前项目中。

> 结语：面试点只是一个表象，你需要针对知识点找到它对应的知识面，等你看完你会发现由点及面的掌握了知识，这样才能举一反三，触类旁通。怎么样收集这个知识点呢？那就是下文讲到的《我的面试经历》中通过试水收集。你可以通过自己试水获得，也可以分析一下笔者的《我的面试经历》中分析**

6. 怎么复习



6.1 收藏 != 拥有

你的微信收藏是不是有很多未读文章？如果你不定期的整理收藏夹，那么等于没有收藏夹。收藏是好习惯，看到了不错的文章就收藏下来，并且微信的订阅推送体系是获取知识开拓眼界最好的方式，但是你需要懂得利用。

笔者每隔一周就会清理一次收藏夹。当然清理不是删除，是把有用的知识点抽离出来，然后通过搜索资料和查阅书籍把他变成自己的知识存储到“有道云笔记”，虽然有道云笔记没有印象笔记稳定，笔者只是喜欢他的 Markdown 功能。

6.2 博客不一定真实

好多人复习的时候都是看一下网上的面经，整理一下知识点，然后百度，谷歌看几篇帖子就以为自己掌握了，自以为已经“学富五车”，其实不是这样的。

网上的文章不一定讲的全面，好多都是以偏概全，甚至有的不一定准确，加上国内的复制，抄袭严重，能看的文章其实是少之又少。所以这里笔者推荐复习一定要看书，原因很简单，写博客的门槛是在太低了，随便动动手就来一篇，然而书不是那么简单，校审，出版社层层把关，而且业界那几本值得看的书大家其实还是知道的。

举一堆栗子，如果你想复习 JVM，那就看《深入理解 Java 虚拟机：JVM 高级特性与最佳实践(第 2 版)》，如果你想复习多线程那就看《Java 并发编程的艺术 (Java 核心技术系列)》，如果你想复习 ElasticSearch 就看《ElasticSearch 权威指南》，如果看 Redis 就看《Redis 实战》，不过笔者也看过国产的一本不错的 Redis 书籍《Redis设计与实现 (数据库技术丛书)》，《高性能 MySQL》，《Spring 实战》，《RabbitMQ实战:高效部署分布式消息队列》等等，当然笔者只是根据这几个领域列举，如果这些不包含你想看的，一定要买一本这个领域经典的书来看，好书永远都不过时。

这里一定要提醒大家，看书一定不要走马观花，我们是从书中汲取知识，不是以量取胜，千万不能有我多久多久就看完一本书而自豪，那样其实你并没有真正掌握里面的知识，小笔者最长看书的时候，第一章就看了一个月（当然是每天看一点）。

原因是每看到一个知识点就不懂了，马上停下来查网上的资料和相关的书籍，如果别的书籍也有讲到就不懂的就继续刨根问题，直到所有问题都解决了再继续看当前书籍的下一个章节。这样以后你觉得你慢了？其实你是快了，因为你掌握了更多的东西，再看接下来的章节其实会很快。

这样坚持看完几本书以后你就会发现自己有些特别了。

6.3 好记性不如烂笔头

what？都什么年代了你还和我说用笔？是的，你没看错，笔者尝试的最好的复习方式就是买一个笔记本，每天把自己看书所得整齐的写上去。网上的云笔记多方便，一粘贴一大堆，但是就是因为他方便才更不容易加深印象，和上面的**《收藏 != 拥有》**一个道理，你以为你整理到云笔记上面了，其实你就是复制粘贴了一下，完全没有过脑。

所以记笔记一定要用纸质的，即便是笔者用云笔记记录，也不会粘贴。还有一个就是感觉，为什么 Kindle 那么火，还那么多人买纸质书呢？

6.4 配合源码

看书一定会有不懂的地方，不理解的点，如果遇到直接看源码就好了，比如 JVM，Redis 都可以找到源码的，虽然他们都是 C 的代码，但是我们有 JAVA 的基础，都不在话下。其他的原本就是 JAVA 实现的看其源码还会更简单，尤其是 JAVA 并发包下面的内容。所以遇到问题一定要仔细的钻研源码。

> 笔者在看 hashcode 的时候就是不理解，为什么每次运行主线程的类 hashcode 不变，其他类会变，于是笔者在 os.cpp 中的 `os::random()` 方法找到了原因，他 random 的 seed 不变，所以每次第一个线程的 hashcode 势必一样。虽然可能和面试不相关，但是起码以后不会因为这个问题而困扰。

6.5 举一反三，触类旁通

学而不思则罔，思而不学则殆。学习以后一定要多总结和思考，比如你知道 JAVA 里面的 PriorityQueue 是优先队列，那么 ElasticSearch 的query 默认搜索数量要限制 2000 也是因为它使用的是分布式的 PriorityQueue 吗？

比如你可以思考 JAVA 的 HashMap 的hash 实现为什么使用 `(h = key.hashCode()) ^ (h >>> 16);`

- 这样一步位运算？
- 它原理和 Redis 的是否一样呢？
- MySQL 在分库分表的时候是否也可以使用类似的原理？
- ElasticSearch 也有分片的概念，那么是不是也用的同一种hash原理？
- hash是不是有多种实现，哪一种更好用，哪一种hash冲突的概率更小？

这个课题答案并不重要，重要的是你在探寻答案过程中的收获。

6.6 学以致用

通过上面的学习想必你已经掌握了一些东西了，那么掌握里以后怎么让他们更有意义？那就是 `Redesign` ，不知道这词好不好理解。意思就是你既然通过分析知识点掌握了之前没有掌握的知识，那么为什么没有在原来的项目中使用呢？知识没有真实的使用场景，面试官怎么认可你？所以通过复习掌握的知识一定要及时的应用到原来的项目上面，具体怎么应用就要看你之前的项目了。

比如你自己学习了 Dubbo，就把原来的项目改成全部使用 RPC 调用的项目；比如你学习了 Spring - Boot，就把原来的 Tomcat 的方式全部去掉；比如你学习了多线程，就去把原来的耗时任务修改掉；比如你学习了设计模式，就思考一下原来什么地方修改一下更优雅？

笔者抛砖引玉举一个自己学以致用的例子，笔者学习了 `Redis` 的数据类型和实现原理以后，觉得有序集合（sorted set）设计很适合做每周最热阅读排行，这是笔者前公司的一个功能接口，于是笔者就使用 sorted set 重新设计了每周最热阅读排行，为什么说他适合呢？

笔者理解到 sorted set 里 items 内容大于 64 的时候同时使用了 hash 和 skiplist 两种设计实现。这也会为了排序和查找性能做的优化。添加和删除都需要修改 skiplist，所以复杂度为 O(log(n))。 但是如果仅仅是查找元素的话可以直接使用 hash，其复杂度为 O(1)，其他的 range 操作复杂度一般为 O(log(n))，当然如果是小于 64 的时候，因为是采用了 ziplist 的设计，其时间复杂度为 O(n)。这样以后查询和更新阅读都变得简单，那么这样简单的就实现了之前复杂逻辑而且效率并不高的接口何乐而不为？

如果你也想初步看一下 Redis 的数据类型和实现原理，可以看看 《Redis 设计与实现 (数据库技术丛书)》，讲的还算是浅显易懂。

> 结语：收藏没有用，重要的是消化吸收，博客只是让我们接触知识的一种途径，但是深入理解这个知识一定要通过看书来完成，看书的同时不要忘记做笔记，掌握知识以后及时使用到项目中。

7. 最高效的投递途径

万事俱备，只欠东风。什么都准备好了，什么途径是最高效的呢？笔者也不敢说能完全了解。

不过笔者整理了下自己的投递方式，以供大家参考。

7.1 拉钩不是很理想

笔者刚整理完上面那份“次的简历”，在拉钩上面投了 10 份，杳无音信，于是笔者思考一下是不是简历质量问题，开始修改，但这是下文，HR 小姐姐就没有点击查收，总不会因为文件名就看不上了？

所以笔者私以为简历没关系，而是拉钩的处理速度很慢。

7.2 内推效果佳

互联网圈子这么小，工作几年你会发现，差不多的公司都有自己的人脉，找他们帮忙推一下就好了。

> 笔者之前很不好意思找熟人帮忙内推，觉得万一挂了，会给熟人丢面子。但是等小白进了一线公司以后发现，内推是一种常态，甚至有的部门要求内推名额，所以不要担心内推会给你的朋友带来负面影响，他很愿意为你效劳。

7.3 订阅号推荐也不错

通过上面笔者的侃侃而谈，你是不是发现自己也有一些订阅的大 V？那么主动的去联系他们一下，询问一下在什么公司高就，能不能帮忙内推，你会得到意想不到的结果。

7.4 别忘记单身会所

Github 也是一个不错的找工作的地方，你每天在逛 Github 的时候发现不错的人记得 follow 一下，时不时看一下他们的动态，发现做了一些你很向往很喜欢的事情，那么事情好办了，说明你未来的同事你很崇拜和欣赏，那么去勾搭一下吧。

7.5 果汁简历

果汁简历 这是笔者最近发现的一个订阅号，每天会推送一些求职攻略，有一些质量还不错，还能免费帮忙大厂内推，这样就不用你一个一个找大厂的订阅号了，也是一个可以选择的途径，笔者强烈推荐。

7.6 Boss 直聘

这个算是笔者最喜欢的了，上面的 Boss 不一定是 HR，好多都是你将来的同事，比如我[阴险脸]。其实想想也对，最了解部门需求和同僚水平的还是同行，那么技术直接来招聘其实也是最有效的方式了，所以 Boss 直聘 是我使用的反馈最快效率最高的招聘平台。笔者强烈推荐。

> 笔者又啰嗦一句，一个平台某个公司挂了，没关系换个平台再投；一个平台某个公司部门关了，没关系，换一个部门再投。笔者的亲身经历，不同平台和部门之前影响很小。下文笔者的面试试水经历有更详细的介绍。
>
> 结语：好的简历很重要，但是投递的途径也很重要，一个高效的途径可以节省时间，也能合理的安排自己的面试节奏。

8. 合理规划自己的面试和复习节奏

这一章节其实内容相对比较少了，笔者放到这里只是为了总结一下，因为面试节奏很重要。下文的章节《我的面试经历》会具体罗列一下面试的节奏和时间节点。

本文主要输一下几点：

1. **面试试水节奏要把握好**，既然开始了就不要轻易的拖延和停止，通常两周面两次，或者一周面两次的节奏比较合适，能对自己的复习马上试错，再次检验和整理。不能拖太久，不然节奏和掌握程度效果很不好，记忆曲线你也是明白的。
2. **体量均衡**，如果你想去一线公司，建议不要用小公司试水，有的朋友担心自己现在没有准备好，想去小公司试试，然后再试大公司。其实没用，小公司的套路和大公司的套路差很多，大部分小公司只需要你过来干活就可以了，面试的知识点大多数偏应用，而大公司多数偏原理和能力，所以如果你想试水，一定拿体量相当的尝试。PS：反正互联网公司这么多。
3. **及时复习**，上面说了很多笔者复习技巧，面试结束后第一时间整理面试点，分析面试点，然后全方面复习。怎么复习呢？那就参照上文的章节《怎么复习》。

9. 面试技巧

9.1 自我介绍

每一次面试都会有一个开场，通常开场会以求职者的自我介绍开始，所以你要准备一个简短却能把经历讲的说明的自我介绍。内容不需要长篇大论介绍项目，简历中不已经有了吗？这是一个开放性的话题，可以快速的让面试官认识你。笔者认为如下几点还蛮重要的，供你参考。

1. 履历，快速的描述一下自己的履历，学历，毕业时间，从事了几份工作，都是什么行业，主要擅长哪方面。这样面试官了解你的情况，也好进一步提问。

2. 业余，工作是你面对公司的，但是你的业余是什么样子的，比如业余时间我会看书，写博客，开源小东西，当然不是让你说我业余时陪女朋友，看电影，而是说一些和技术相关的。这样让面试官可以看到“工作以外”的你。

3. 原因，每个面试官还是蛮关心你的离职原因，所以你开篇直接简单介绍一下你的情况和离职原因，这样让面试官了解了情况。

9.2 知之为知之

好多人听闻有这样的面试经验，如果遇到问题不会以后找一个自己会的点侃侃而谈，这样不仅能盖住刚才自己的不会，还能展示自己另一面的实力。这种想法是错误，知之为知之，不知为不知，如果你不会直接干脆的说，这个地方我不了解，这个我不清楚，这个我没有遇到过。

原因很简单，面试官阅人无数，这些套路他早知道，反而觉得你不诚恳，答非所问的浪费彼此时间。面试官都会考一些通用的基础问题，但是也会贴近当前项目中使用的技术来提问，因为大部分面试官不是来选拔培训者，而是选拔干活的人，所以他们需要知道你当前掌握技能的程度，不要错用技巧。

9.3 最有挑战的项目

好多面试官都喜欢问这个问题，他想知道你处理问题的难易程度和你处理问题的方式，当然这个问题也有其他的提问方式，比如“你遇到过比较困难的问题吗，怎么解决的”。

所以这个地方需要我们提前准备的，当然不是让你准备好说谎，而是面试之前仔细推敲自己的经历，找出一些自己觉得有挑战的事情。可以是业务上面的难点，技术上面的突破，黑魔法实现的功能，实际一次内存泄漏的调试，每一个点都能说明你的工作难度。

9.4 一言不合就上图

没有绝对的权威，如果面试过程中觉得面试官说的不恰当，或者是自己怎么也描述不清楚，那么直接找一个白板画图说一下自己的思路。笔者参加的几个面试基本都有白板，还专门准备了白板笔。所以对于自己参与的项目的架构图，自己学习的中间件的流程图，系统的例图等，还是要有充分的理解。

同时呢面试的时候不要怯场，哪怕是问你一个 CurrentHashMap 的原理，如果你愿意展示自己的备课能力，也可以去白板上面画一画。

9.5 备课

选择是双向的，所以大多数面试官问完你问题以后都会问你，你有什么要问我的？这个时候你的主场就来了，虽然是让你问他，但是这是最好的机会让你自由发挥的时候。

你可以提前插好公司和部门的背景和市场的前景，结合自己的思考问一下当前业务的发展。也可以提前看一下技术点，团队合作，开发方式等具体的问题，一方面你可以了解这个公司的模式是不是你喜欢的，另一方面面试官也会觉得你是一个细心的人。

当然你不要误解笔者是故意让你准备一些什么，而是让你准备好这些面试官可能问的问题和经历的面试环节需要的点，别突然遇到了让自己无计可施，并非刻意而为。

9.6 真实

面试过程中难免会有一些除了技术以外的话题，比如你怎么看待这种情况，你怎么怎么对待这种人，遇到这种情况你怎么办？有一些面试技巧的人一下子就猜出来面试官的意图，故意按照面试官的意思回答。

笔者私以为这样其实不对，你就完全按照自己的想法去说就好了，如果你因为猜到了面试官的想法因此获得了 offer，那么你每天工作的环境可能都需要伪装的你，时间久了你也会觉得那并不适合自己。

面试其实是双向的选择，面试官选择你，同时你也在选择公司。

> 结语：真实是最好的面试技巧，知之为知之，不知为不知，是知也。但是也要提前准备面试过程中可能遇到的问题。

10. 面试通过怎么从容的谈 offer

10.1 不要轻易说出自己的底线

这是你的底牌，有个谈判原则，先亮底牌，往往会失去主动权，面试完一轮就问你期望薪水，这时你可以回答给个范围：如他的岗位招人薪水范围椒是 15k - 30k，你也可以回答：我期待薪水是：15k - 30k。如果他再问你具体的话，你可以说：现在讨论这个还过早，我相信在决定录用我前，再讨论会更合适。这样大家都不会尴尬。

10.2 和 HR 沟通要大胆说出自己的期望

一定要说出自己的期望，如果你到了 HR 环节，一个公司的招聘成本真的太高了，如果你真的到 HR 环节，那么说明你足够优秀。足够优秀 HR 就不会因为你要的太高拒绝你，当然你要的太离谱他肯定会给你一个最大的价格，你自己要控制好就可以了。

10.3 不后悔

来之前谈好待遇，来之后无论多少都不能后悔，这是底线。既然选择便只顾风雨兼程，不然你工作也没心情，没动力，最后害得不还是自己?

11. 你想要的“我的面试经历”

时隔比较久，面试问题笔者就记下了一些重点，整理给大家，仅供参考。

排名是时间先后

11.1 瓜子二手车

1. 快排；
2. 二分法查找，推到时间复杂度；
3. BlockingQueue 实现原理；
4. 二叉树遍历；
5. ElasticSearch 存储结构和原理；
6. MySQL 索引，优化；
7. SpringMVC 请求过程；
8. Dubbo 原理。

11.2 爱奇艺

1. SpringMVC 请求过程；
2. Dubbo 原理；
3. MySQL 缓存，性能优化，索引结构；
4. ElasticSearch 存储结构和原理；

5. 设计模式；
6. Proxy 和 CGLib 区别和原理；
7. 乐观锁和悲观锁；
8. ThreadLocal；
9. 闭包。

11.3 每日优鲜

1. TCP 三次握手；
2. MySQL 分库分表，性能优化；
3. HashMap 实现原理；
4. 库存设计原理(电商相关)；
5. Redis 基本数据结构和使用；
6. ElasticSearch 存储结构，查询过程，索引原理；
7. 多线程；
8. 生产者消费者。

11.4 滴滴

1. 闭包；
2. 反射的实现和原理；
3. JDK8 lambda 使用和原理；
4. SQL 注入和 XSS 攻防；
5. 设计模式；
6. Redis 基本数据结构和原理；
7. Http1.0 和 Http2.0 区别；
8. MySQL 索引结构，基本数据类型和使用场景；
9. Shell 分析日志查看 QPS；
10. JDK8 中多线程的变化；

11.5 京东

1. Spring AOP 原理，Proxy 和 GCLib 原理；
2. 多线程；
3. CountDownLatch；
4. 库存设计；
5. 大秒设计；
6. SpringMVC 请求过程和使用设计模式；
7. 手写单例；
8. Redis sorted set 实现原理；
9. HashMap 实现原理；
10. MySQL 索引和优化；
11. 分布式扩容和容灾；

12. PriorityQueue 使用和原理；

11.6 阿里巴巴

1. HashMap 和 CurrentHashMap 区别；
2. Hashcode；
3. 用过的集合类；
4. ThreadExecutorPool ；
5. volatile 内存模型；
6. ElasticSearch 集群路由规则和容错；
7. Dubbo 原理；
8. Redis Hash 设计原理；

12. 彩蛋，终于等到你

终于等到你，还好没放弃。

如果觉得笔者写的对你有帮助，欢迎订阅笔者的微信订阅号「码匠笔记」，持续关注笔者和给予指点。当然这不是彩蛋哦。彩蛋是阿里巴巴新零售诚聘 P6 ， P7，坐标北京，杭州，领导 nice，发展空间大，薪资丰厚，运气好还有股票。如果你有兴趣，欢迎关注笔者「码匠笔记」订阅号，留言「内推」，笔者会第一时间反馈。

> 当然笔者也需要质量的简历哦，请仔细斟酌自己的工作经历和修改简历，充分准备以后，欢迎发送 PDF 简历给笔者。

13. 最后

已到文末，万分感谢。笔者码字不易，如果您觉得这篇文章对您有帮助，欢迎转发给需要的人；如果您觉得这篇文章有需要改进的地方，欢迎留言指正。