**作者:***[hsm\_computer](https://www.cnblogs.com/JavaArchitect/)*

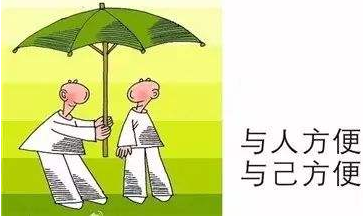
# **地址：https://www.cnblogs.com/JavaArchitect/p/10011253.html**

# **[最近面试java后端开发的感受：如果就以平时项目经验来面试，通过估计很难——再论面试前的准备](https://www.cnblogs.com/JavaArchitect/p/10011253.html)**

    在上周，我密集面试了若干位Java后端的候选人，工作经验在3到5年间。我的标准其实不复杂：第一能干活，第二Java基础要好，第三最好熟悉些分布式框架，我相信其它公司招初级开发时，应该也照着这个标准来面的。

    我也知道，不少候选人能力其实不差，但面试时没准备或不会说，这样的人可能在进团队干活后确实能达到期望，但可能就无法通过面试，但面试官总是只根据面试情况来判断。

    但现实情况是，大多数人可能面试前没准备，或准备方法不得当。要知道，我们平时干活更偏重于业务，不可能大量接触到算法，数据结构，底层代码这类面试必问的问题点，换句话说，面试准备点和平时工作要点匹配度很小。



    作为面试官，我只能根据候选人的回答来决定面试结果。不过，与人方便自己方便，所以我在本文里，将通过一些常用的问题来介绍面试的准备技巧。大家在看后一定会感叹：只要方法得当，准备面试第一不难，第二用的时间也不会太多。

1 **框架是重点，但别让人感觉你只会山寨别人的代码**

    在面试前，我会阅读简历以查看候选人在框架方面的项目经验，在候选人的项目介绍的环节，我也会着重关注候选人最近的框架经验，目前比较热门的是SSM。

    不过，一般工作在5年内的候选人，大多仅仅是能“山寨”别人的代码，也就是说能在现有框架的基础上，照着别人写的流程，扩展出新的功能模块。比如要写个股票挂单的功能模块，是会模仿现有的下单流程，然后从前端到后端再到数据库，依样画葫芦写一遍，最多把功能相关的代码点改掉。



    其实我们每个人都这样过来的，但在面试时，如果你仅仅表现出这样的能力，就和大多数人的水平差不多了，在这点就没法体现出你的优势了。

    我们知道，如果单纯使用SSM框架，大多数项目都会有痛点。比如数据库性能差，或者业务模块比较复杂，并发量比较高，用Spring MVC里的Controller无法满足跳转的需求。所以我一般还会主动问：你除了依照现有框架写业务代码时，还做了哪些改动？

     我听到的回答有：增加了Redis缓存，以避免频繁调用一些不变的数据。或者，在MyBitas的xml里，select语句where条件有isnull，即这个值有就增加一个where条件，对此，会对任何一个where增加一个不带isnull的查询条件，以免该语句当传入参数都是null时，做全表扫描。或者，干脆说，后端异步返回的数据量很大，时间很长，我在项目里就调大了异步返回的最大时间，或者对返回信息做了压缩处理，以增加网络传输性能。

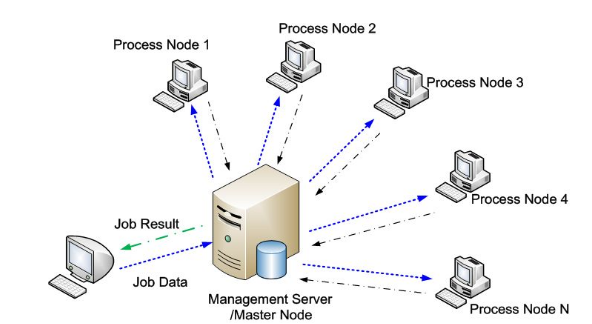
    对于这个问题，我不在乎听到什么回答，我只关心回答符不符逻辑。一般只要答对，我就会给出“在框架层面有自己的体会，有一定的了解”，否则，我就只会给出“只能在项目经理带领下编写框架代码，对框架本身了解不多”。

    其实，在准备面试时，归纳框架里的要点并不难，我就不信所有人在做项目时一点积累也没，只要你说出来，可以说，这方面你就碾压了将近7成的竞争者。



**2 别单纯看单机版的框架，适当了解些分布式**

    此外，在描述项目里框架技术时，最好你再带些分布式的技术。下面我列些大家可以准备的分布式技术。



    1 反向代理方面，nginx的基本配置，比如如何通过lua语言设置规则，如何设置session粘滞。如果可以，再看些nginx的底层，比如协议，集群设置，失效转移等。

    2 远程调用dubbo方面，可以看下dubbo和zookeeper整合的知识点，再深一步，了解下dubbo底层的传输协议和序列化方式。

    3 消息队列方面，可以看下kafka或任意一种组件的使用方式，简单点可以看下配置，工作组的设置，再深入点，可以看下Kafka集群，持久化的方式，以及发送消息是用长连接还是短拦截。

    以上仅仅是用3个组件举例，大家还可以看下Redis缓存，日志框架，MyCAT分库分表等。准备的方式有两大类，第一是要会说怎么用，这比较简单，能通过配置文件搭建成一个功能模块即可，第二是可以适当读些底层代码，以此了解下协议，集群和失效转移之类的高级知识点。

    如果能在面试中侃侃而谈分布式组件的底层，那么得到的评价就会比较好了，比如“深入了解框架底层”，或“框架经验丰富”，这样就算去面试架构师也行了，更何况是高级开发。



**3 数据库方面，别就知道增删改查，得了解性能优化**

    在实际项目里，大多数程序员用到的可能仅仅是增删改查，当我们用Mybatis时，这个情况更普遍。不过如果你面试时也这样表现，估计你的能力就和其它竞争者差不多了。

    这方面，你可以准备如下的技能。

    1 SQL高级方面，比如group by, having，左连接，子查询（带in），行转列等高级用法。

    2 建表方面，你可以考虑下，你项目是用三范式还是反范式，理由是什么？

    3 尤其是优化，你可以准备下如何通过执行计划查看SQL语句改进点的方式，或者其它能改善SQL性能的方式（比如建索引等）。

    4 如果你感觉有能力，还可以准备些MySQL集群，MyCAT分库分表的技能。比如通过LVS+Keepalived实现MySQL负载均衡，MyCAT的配置方式。同样，如果可以，也看些相关的底层代码。

    哪怕你在前三点表现一般，那么至少也能超越将近一般的候选人，尤其当你在SQL优化方面表现非常好，那么你在面试高级开发时，数据库层面一定是达标的，如果你连第四点也回答非常好，那么恭喜你，你在数据库方面的能力甚至达到了初级架构的级别。



**4 Java核心方面，围绕数据结构和性能优化准备面试题**

    Java核心这块，网上的面试题很多，不过在此之外，大家还应当着重关注集合（即数据结构）和多线程并发这两块，在此基础上，大家可以准备些设计模式和虚拟机的说辞。

    下面列些我一般会问的部分问题：

    1 String a = "123"; String b = "123"; a==b的结果是什么？ 这包含了内存，String存储方式等诸多知识点。

    2 HashMap里的hashcode方法和equal方法什么时候需要重写？如果不重写会有什么后果？对此大家可以进一步了解HashMap（甚至ConcurrentHashMap）的底层实现。

    3 ArrayList和LinkedList底层实现有什么差别？它们各自适用于哪些场合？对此大家也可以了解下相关底层代码。

    4 volatile关键字有什么作用？由此展开，大家可以了解下线程内存和堆内存的差别。

    5 CompletableFuture，这个是JDK1.8里的新特性，通过它怎么实现多线程并发控制？

    6 JVM里，new出来的对象是在哪个区？再深入一下，问下如何查看和优化JVM虚拟机内存。

    7 Java的静态代理和动态代理有什么差别？最好结合底层代码来说。

    通过上述的问题点，我其实不仅仅停留在“会用”级别，比如我不会问如何在ArrayList里放元素。大家可以看到，上述问题包含了“多线程并发”，“JVM优化”，“数据结构对象底层代码”等细节，大家也可以举一反三，通过看一些高级知识，多准备些其它类似面试题。

    我们知道，目前Java开发是以Web框架为主，那么为什么还要问Java核心知识点呢？我这个是有切身体会的。

    之前在我团队里，我见过两个人，一个是就会干活，具体表现是会用Java核心基本的API，而且也没有深入了解的意愿（估计不知道该怎么深入了解），另一位平时专门会看些Java并发，虚拟机等的高级知识。过了半年以后，后者的能力快速升级到高级开发，由于对JAVA核心知识点了解很透彻，所以看一些分布式组件的底层实现没什么大问题。 而前者，一直在重复劳动，能力也只一直停留在“会干活”的层面。

     而在现实的面试中，如果不熟悉Java核心知识点，估计升高级开发都难，更别说是面试架构师级别的岗位了。



**5 Linux方面，至少了解如何看日志排查问题**

    如果候选人能证明自己有“排查问题”和“解决问题”的能力，这绝对是个加分项，但怎么证明？



    目前大多数的互联网项目，都是部署在Linux上，也就是说，日志都是在Linux，下面归纳些实际的Linux操作。

    1 能通过less命令打开文件，通过Shift+G到达文件底部，再通过?+关键字的方式来根据关键来搜索信息。

    2 能通过grep的方式查关键字，具体用法是, grep 关键字 文件名，如果要两次在结果里查找的话，就用grep 关键字1 文件名 | 关键字2 --color。最后--color是高亮关键字。

    3 能通过vi来编辑文件。

    4 能通过chmod来设置文件的权限。

    当然，还有更多更实用的Linux命令，但在实际面试过程中，不少候选人连一条linux命令也不知道。还是这句话，你哪怕知道些很基本的，也比一般人强了。

**6 通读一段底层代码，作为加分项**

    如何证明自己对一个知识点非常了解?莫过于能通过底层代码来说明。我在和不少工作经验在5年之内的程序员沟通时，不少人认为这很难？确实，如果要通过阅读底层代码了解分布式组件，那难度不小，但如果如下部分的底层代码，并不难懂。



    1 ArrayList,LinkedList的底层代码里，包含着基于数组和链表的实现方式，如果大家能以此讲清楚扩容，“通过枚举器遍历“等方式，绝对能证明自己。

    2 HashMap直接对应着Hash表这个数据结构，在HashMap的底层代码里，包含着hashcode的put，get等的操作，甚至在ConcurrentHashMap里，还包含着Lock的逻辑。我相信，如果大家在面试中，看看而言ConcurrentHashMap，再结合在纸上边说边画，那一定能征服面试官。

    3 可以看下静态代理和动态代理的实现方式，再深入一下，可以看下Spring AOP里的实现代码。

    4 或许Spirng IOC和MVC的底层实现代码比较难看懂，但大家可以说些关键的类，根据关键流程说下它们的实现方式。

    其实准备的底层代码未必要多，而且也不限于在哪个方面，比如集合里基于红黑树的TreeSet，基于NIO的开源框架，甚至分布式组件的Dubbo，都可以准备。而且准备时未必要背出所有的底层（事实上很难做到），你只要能结合一些重要的类和方法，讲清楚思路即可（比如讲清楚HashMap如何通过hashCode快速定位）。

    那么在面试时，如何找到个好机会说出你准备好的上述底层代码？在面试时，总会被问到集合，Spring MVC框架等相关知识点，你在回答时，顺便说一句，“我还了解这块的底层实现”，那么面试官一定会追问，那么你就可以说出来了。

     不要小看这个对候选人的帮助，一旦你讲了，只要意思到位，那么最少能得到个“肯积极专业“的评价，如果描述很清楚，那么评价就会升级到“熟悉Java核心技能（或Spring MVC），且基本功扎实”。要知道，面试中，很少有人能讲清楚底层代码，所以你抛出了这个话题，哪怕最后没达到预期效果，面试官也不会由此对你降低评价。所以说，准备这块绝对是“有百利而无一害”的挣钱买卖。

**7 一切的一切，把上述技能嵌入到你做过的项目里**

在面试过程中，我经常会听到一些比较遗憾的回答，比如候选人对SQL优化技能讲得头头是道，但最后得知，这是他平时自学时掌握的，并没用在实际项目里。

    当然这总比不说要好，所以我会写下“在平时自学过SQL优化技能”，但如果在项目里实践过，那么我就会写下“有实际数据库SQL优化的技能”。大家可以对比下两者的差别，一个是偏重理论，一个是直接能干活了。其实，很多场景里，我就不信在实际项目里一定没有实践过SQL优化技能。

    从这个案例中，我想告诉大家的是，你之前费了千辛万苦（其实方法方向得到，也不用费太大精力）准备的很多技能和说辞，最后应该落实到你的实际项目里。

    比如你有过在Linux日志里查询关键字排查问题的经验，在描述时你可以带一句，在之前的项目里我就这样干的。又如，你通过看底层代码，了解了TreeSet和HashSet的差别以及它们的适用范围，那么你就可以回想下你之前做的项目，是否有个场景仅仅适用于TreeSet？如果有，那么你就可以适当描述下项目的需求，然后说，通过读底层代码，我了解了两者的差别，而且在这个实际需求里，我就用了TreeSet，而且我还专门做了对比性试验，发现用TreeSet比HashSet要高xx个百分点。

    请记得，“实践经验”一定比“理论经验”值钱，而且大多数你知道的理论上的经验，一定在你的项目里用过。所以，如果你仅仅让面试官感觉你只有“理论经验”，那就太亏了。



**8 小结：本文更多讲述的准备面试的方法**

    本文给出的面试题并不多，但本文并没有打算给出太多的面试题。从本文里，大家更多看到的是面试官发现的诸多候选人的痛点。

    本文的用意是让大家别再重蹈别人的覆辙，这还不算，本文还给出了不少准备面试的方法。你的能力或许比别人出众，但如果你准备面试的方式和别人差不多，或者就拿你在项目里干的活来说事，而没有归纳出你在项目中的亮点，那么面试官还真的会看扁你。

    本文里提到的方法和技能，如果能对大家有所帮助，请大家帮忙转发，或者点击下面的按钮来“推荐本文”，或者通过评论来参与讨论。

    本文欢迎转载，不过请注明文章来源，如果可以，请同时给出本人写的两本书的连接[Java Web轻量级开发面试教程](https://item.jd.com/12136095.html" \o "科技IT爆款好书，全场每满200减100，为新学期助力加油！戳我》》" \t "https://www.cnblogs.com/JavaArchitect/p/_blank)和[Java核心技术及面试指南](https://item.jd.com/12421187.html" \o "一本书讲透Java语言核心概念、编程思想、开发实战及面试技巧" \t "https://www.cnblogs.com/JavaArchitect/p/_blank)。

    再次感谢大家读完本文。