# 参考

https://gitee.com/shishan100/Java-Interview-Advanced#https://gitee.com/link?target=https%3A%2F%2Fmp.weixin.qq.com%2Fs%3F\_\_biz%3DMzU0OTk3ODQ3Ng%3D%3D%26mid%3D2247484645%26idx%3D1%26sn%3D663238af983603a4c0b33cf42c3ebbcf%26chksm%3Dfba6ece6ccd165f0d78f271a21fdc1b91ad8d3def896e2f4df79d9c8d4cf99e3b2978d703dde%26mpshare%3D1%26scene%3D1%26srcid%3D0608edSfhNw7AIjzl9R54Sih%2523rd

# 一、备战面试

## **开放式问题：如果让你设计一个消息中间件，你会怎么做？**

背后：让你从整体架构，核心流程，数据结构，等各个层面来考虑，你会如何完成这个设计？

1. **生产消费模型及核心数据结构**

可以先允许数据写入内存作为一个缓冲，然后每隔几秒再把数据刷入磁盘文件中？数据刷入磁盘文件之后，这个磁盘文件有多少个？

按照什么样的规则对磁盘文件做一个拆分？

数据写入磁盘文件之后，有相应metadata来标识这个数据的具体信息？比如这个数据的offset偏移量，或者是一个内置的唯一id？

接着现在数据是被存储在磁盘文件里了，那么此时你如何把数据投递到下游的消费者里去呢？

你的消费模型是什么样的？比如说一个queue里的数据，是会均匀分配给消费者的各个实例呢？还是会怎么做呢？

**// 建议大家可以去研究研究比如kafka底层的文件存储原理，那是非常经典的高性能高并发消息中间件存储架构的实现。**

1. **支撑TB级数据写入的分布式架构**

消息中间件肯定会遇到每天TB级海量数据高并发高吞吐写入的场景。

**你的数据是不是要分布式的存储**？

1. 考虑把一个大的数据集合做分片存储，比如说分成N片数据，每个数据分片放在一台机器上， 这样就可以充分利用多台机器的资源来承载TB级的大量数据了。

2）数据分片是不是要可以支撑扩容？比如你一开始设置的分片数量是10个，存在10台机器上。 结果现在发现10台机器都扛不住了，需要扩容到20个分片，放在20台机器上才可以。

3）支持数据分片的扩容以及自动数据负载均衡迁移？也就是10个分片的数据自动均匀分配给 扩容后的20个分片。

**// kafka在这块的架构设计，非常的优秀，采用了partition的概念实现数据分片，支持分布式的数据存储，而且还支持动态扩容。**

1. **数据宕机场景下的高可用架构**

一般分布式系统实现高可用架构，都是采用多副本冗余机制（参考Kafka）

1. **支持数据不丢失的ack机制**

必须考虑两块ack机制，

1）生产端，一旦投递了消息，必须要求他将数据比如写入多个副本之后，才返回一个ack回调响应。否则要是一直没收到ack的话，就需要重发一条消息过去，保证生产投递成功。

1. 消费端，一旦消费处理成功一条消息了，必须返回一个ack给消息中间件，然后消息中间件才能删除这条消息。否则一旦消费者宕机，就必须重发这条消息给其他的消费者实例，保证消息一定会被处理成功。

**// kafka：基于rabbitmq来阐述的这个数据不丢失的全链路ack机制。**

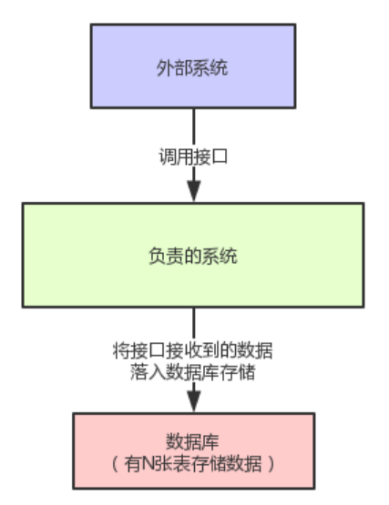
## 阿里三面，P9面试官是如何360°无死角考察候选人的？

*对自己负责系统的架构演进完全无意识，负责系统的难点从没思考过，系统涉及的一些技术的细节不了解，没有技术深度的积累，都导致他在三面表现很不好，最后就是直接挂掉。*

**之前实现的系统：**

每天都会有人调用他的接口，然后有数据会落入数据库表中。

这个系统每天接口调用大概会落入数据库中有20万左右的数据量，那么每个月大概是600万左 右的数据量，每年大概是近亿级的数据量会落入数据库中。

****

### 架构演进考察

第一个问题：如果假设你的系统承载的业务量翻了10倍，每天新增200万数据，你的系统架构要如何演进？

**需要引入分库分表的技术**，保证每个库每个表分散一定的数据量，避免单表单库数据量过大。

**如果100倍增长呢，每年单表新增近10亿数据**，你分库分表也不一定够了。因为此时可能会有高并发访问的问题，数据库抗起来很吃力。

**你要不要考虑数据异构、冷热分离等数据存储的架构设计？**

比如采用MySQL分库分表 + 分布式NoSQL数据库 + Elasticsearch分布式搜索 + Redis缓存的架构，来整体设计这个数据存储架构。

**1、你可以先做冷热分离的架构，**比如最热的数据放入分布式NoSQL数据库，专门承载当日数据的高并发写入，以及高性能的读写。然后每过一段时间，做数据归档，把NoSQL里不再频繁使用的冷数据迁移到MySQL里去归档。

**2、应对海量数据的检索**，可以把索引构建在Elasticsearch里来应对，但是从NoSQL+MySQL的异构存储来提取明细数据即可。

**3、针对一些特别热查询的数据，可以依托Redis做一个缓存。**

其实那个P9面试官的面试评价里，期望的也是候选人把这一套架构说出来。虽然P6+的职级不一定说有能力完全hold住这个架构，但是起码要有这个意识。

### 对公司底层技术的原理考察

你能说说你对这种特殊存储服务的理解吗，他的原理是什么？

你们用的云平台上的服务存储他的架构是什么样的，你们的存储是如何规划的？

### 系统难点的考察

举个例子，比如上面的这个系统，实际上他有一个步骤是要做数据迁移，也就是说把数据库里可能几百万数据量，一次性迁移到另外一套存储里去

那么这个数据迁移的步骤，其实涉及到千万级的数据量迁移。

你如何保证数据迁移的效率？如何保证迁移后的数据准确性？在迁移的过程中如何避免影响数据库的性能？

### 擅长技术的考察

大公司一定会考察你的技术深度，一般就是对你平时用的最多，或者最熟悉的技术深入挖掘和考察，看你的技术深度有多深。比如MQ

## 互联网公司的面试官是如何360°无死角考察候选人的？（上篇）

### ****真实的面试连环炮****

将会从消息中间件、分布式缓存、分布式搜索、分布式架构、海量数据、NoSQL、高并发、高可用、高性能、数据库、JVM虚拟机等各方面聊一下面试中的高频技术问题。

比如说看你简历上写了**熟悉消息中间件（MQ技术）**。可能会有一个下面的连环炮式发问：

* 说说你们公司线上生产环境用的是什么消息中间件？
* 那你们线上系统是有哪些技术挑战，为什么必须要在系统里引入消息中间件？
* 你们的消息中间件技术选型为什么是RabbitMQ？
* 为什么不用RocketMQ或者是Kafka？技术选型的依据是什么？
* 你们怎么保证消息中间件的高可用性？避免消息中间件故障后引发系统整体故障？
* 使用消息中间件技术的时候，你们怎么保证投递出去的消息一定不会丢失？
* 你们怎么保证投递出去的消息只有一条且仅仅一条，不会出现重复的数据？
* 如果消费了重复的消息怎么保证数据的准确性？
* 你们线上业务用消息中间件的时候，是否需要保证消息的顺序性？
* 如果不需要保证消息顺序，为什么不需要？假如我有一个场景要保证消息的顺序，你们应该如何保证？
* 下游消费系统如果宕机了，导致几百万条消息在消息中间件里积压，此时怎么处理？
* 你们线上是否遇到过消息积压的生产故障？如果没遇到过，你考虑一下如何应对？
* 你们用的是RabbitMQ？那你说说RabbitMQ的底层架构原理，逻辑架构、物理架构以及数据持久化机制？
* 你们RabbitMQ的最高峰QPS每秒是多少？线上如何部署的，部署了多少台机器，机器的配置如何？
* 你们用的是Kafka？那你说说Kafka的底层架构原理，磁盘上数据如何存储的，整体分布式架构是如何实现的？
* 再说说Kafka是如何保证数据的高容错性的？零拷贝等技术是如何运用的？高吞吐量下如何优化生产者和消费者的性能？
* 看过Kafka的源码没有。如果看过，说说你对Kafka源码的理解？
* 你们用的是RocketMQ？RocketMQ很大的一个特点是对分布式事务的支持，你说说他在分布式事务支持这块机制的底层原理？
* RocketMQ的源码看过么，聊聊你对RocketMQ源码的理解？
* 如果让你来动手实现一个分布式消息中间件，整体架构你会如何设计实现？

### ****技术广度的考察****

工程师**首先要避免自己的技术短板。**整体技术栈要或多或少都知道一些，不能出现盲区。

你们公司有没有什么业务场景是可以用NoSQL的？现在国内各个公司用NoSQL的技术都有哪些选型？具体NoSQL可以解决什么问题？

因此，上面说的消息中间件、分布式缓存、海量数据、分布式搜索、NoSQL、分布式架构、高并发、高可用、高性能这些技术。并不是说真的要求工作几年的同学都要精通到源码层面。

### ****底层技术的考察****

现在很多互联网大厂都会有基本功的考察，

Java虚拟机核心原理、内存模型、垃圾回收、线上FullGC卡顿优化、线上OOM问题。

Java并发中的volatile、锁优化、AQS源码；

Netty背后的IO、网络相关的知识。

// 其实这种底层技术，是线上高负载大型系统的架构设计和开发，必须要具备的。

因为底层技术不扎实，很多中间件或其他高阶的技术，都无法深入理解其原理。

而且很多时候，解决线上系统的生产故障，都需要这些技术。因此，底层技术的掌握是一个优秀工程师必须具备的素养。

### ****技术深度的考察****

**比如项目用了Redis或者Elasticsearch。只要你用过了，而且是你某个项目里的核心技术，那么一定会用连环炮式的发问，深入各种细节、底层、生产环境可能遇到的技术挑战。**

**总之，就是要用压力测试出来你在这块技术水平掌握的到底有多深，实践经验有多强。**

**举个例子**

* 比如说你对一个技术的掌握是否达到了源码级别？
* 是否对某个框架，或者是中间件深入的理解底层的源码实现，从源码级别说清楚他的架构原理？
* 是否对这个技术有过线上的高可用部署，承载过高并发流量的访问？
* 是否对这个技术在线上生产环境解决过各种各样的复杂技术挑战？
* 是否基于这个技术落地到你的业务系统中，设计出各种复杂的系统架构？
* 技术深度的考察是中大型互联网公司面试官对一个**高级/资深**的候选人必须考察的。

因为如果一个人工作5年以上，来应聘高级职位的话，那我们绝对是要求他对至少一个技术领域有着较为深入的研究的。

**技术广度决定了你可以利用各种技术来做项目，但是技术深度决定了你的技术功底。**

你未来学新东西有多快，线上系统出了故障你能否快速定位和解决，你能否基于对技术的深刻理解为公司的项目设计和开发出复杂而且优秀的架构出来，这都取决于技术深度。

## 互联网公司面试官是如何360°无死角考察候选人的？（下篇）

### ****项目经验的考察****

画一下项目整体架构图

从项目入手，考察你项目里各个技术掌握的如何，通过连环炮对你掌握最好的技术进行深入考察，对一些高阶技术的考察，直接下探到底层。

举个例子，公司用dubbo作为服务框架，那么dubbo底层的通信框架是什么？Netty？Mina？

然后底层NIO是啥？网络通信里的长连接和短连接是啥？

你是否看过dubbo的源码？dubbo源码中你印象深刻的对并发技术的运用是什么？

提问时，必须要深入到你把某个业务细节讲清楚，以及结合这个业务细节到底是如何落地和设计技术方案的，如何使用各种技术在业务中的。

**● 你说你用了Redis，那就会进一步问你，你哪个业务用了Redis？那个业务的流程请你叙述 一下？**

● **在Redis里你们具体是选用了哪种数据结构存放什么数据？数据的过期时间是什么？如果 缓存过期了，你的数据兜底方案是什么，到哪儿去回查？**

● **你的key如何设计的，为什么要这么设计？你的这个业务把数据放在了Redis里，是其他 哪个业务来查Redis？为什么要这样子做？如果不用Redis会怎么样？**

实际上各种技术都可以在项目里深扣细节。你对这个技术的实践到底有多深，经历过多么复杂的线上业务的实践，能hold住一个技术解决线上系统中的哪些问题。

从项目里，

可以看出是否负责过复杂业务架构下的分布式系统的设计和开发？

系统是否是线上高并发大流量高负载场景的挑战，你是否经历过这种技术挑战？

系统是否承载过亿级别海量数据的存储以及高性能读写的挑战，你是否解决过这些问题？

看出你的整体能力技术定位。你是仅仅负责过一个模块呢？还是负责过一个子系统？

或者作为架构师负责过一个完整项目群，带过几十人的团队，设计过大规模复杂的系统架构？

**中级的岗位**，觉得你技术整体ok，独立负责过核心模块的开发，同时对各种技术都有一定的实践经验

**高级/资深的岗位：**是否带领一个小团队独立负责过有一定复杂度和难度的完整系统架构设计和开发。

**架构师的岗位：**要求你在一个公司里主导过很多人协作完成的大型而且复杂的项目群。

要求你对一个大型系统架构有深度的思考和整体的把控，而且这个项目要有足够的技术挑战，大用户量、高并发、海量数据，等等。

*因此，项目考察，是重中之重。很多同学平时积累了不少的技术学习，但是有一个很大的问题是，项目经验和实践太少。这些同学可能确实没经历过复杂系统的架构设计的历练，所以非常容易在项目经验考察这块出现问题，被面试官判定为技术不错，但是经验缺乏。*

### ****系统设计的考察****

一般会用自己公司或者团队里的一些业务场景拿出来，或普遍性的一些业务场景，然后问你如何针对这个业务场景设计系统架构？

**举几个例子**：

如何设计一个电商秒杀系统架构？

如何设计一个消息推送系统架构？

双11大促的时候如何设计系统的动态扩容/缩容的机制？

类似诸如此类的一些场景式的系统设计考察。其实这个主要是用一些你可能没接触过的场景，来现场考察一下你的架构设计思维。尤其是针对上面说的高级/资深、架构类的岗位，我们尤其会注重现场考察你没接触过的业务场景的架构设计。毕竟你来了以后，肯定要让你接触全新的业务，然后立马给出合理而且靠谱的架构设计方案，在新的公司来落地你的经验。

**所以，平时应该对公司里各种业务场景多思考，自己设定一些挑战，比如假设你公司的请求量暴增100倍，数据量暴增100倍，你的系统架构应该如何设计？**

**多给自己设立挑战，然后去尝试着思考设计，才能积累出系统设计的思维和能力来。**

### ****基本功的考察****

### ****多轮面试官的分工协作****

**一面面试官**可能主要就是考察一下技术内容，包括技术面以及连环炮发问考察技术深度，以及算法功底，不太涉及项目。

**二面面试官**可能会着重考察项目经验，系统设计，同时对技术深度也会继续考察。

**三面面试官**可能会从你把控过的项目规模、带的团队规模、团队管理能力、规范和流程设计能力、整体工作履历背景和经验、软素质（沟通表达、团队协作、价值观，等等）来考察你。

## 中小公司的Java工程师应该如何逆袭冲进BAT？

### ****小公司从业和困境****

一直徘徊在各种中小公司里开发一些没技术难度的Java系统，主要就是CRUD。

用了MQ、缓存、分库分表，但是也没什么并发量，数据量也不算特别大，自己的技术成长极为缓慢。

三五年，七八年，甚至十多年，职业发展和技术水平都停滞在这个状态，无法有更进一步的发展。

随着现在寒冬到来，到处裁员，中年码农的危机，加不动班，体力越来越差，孩子压力越来越大，对自己何去何从很迷茫。

### ****你的技术为啥十年八年都无法进步？****

拆解一下，你的能力集中在哪几块：

**技术广度**

对MQ、缓存、NoSQL、大数据、高并发、高可用、微服务，等一系列的相关技术都有一定的 了解，熟悉常见功能

在自己的项目里落地使用过，有一定的技术使用经验

**技术深度**

读过Kafka的底层源码？

对消息中间件的架构设计思想有深刻的理解？

对分布式事务框架/中间件的架构设计有过研究？

在每秒百万并发场景下做过底层系统的深入优化和故障处理？

**项目经验**

你有没有整体负责过几亿注册用户，几千万日活用户的大规模、高并发、分布式、高可用、高 复杂度的系统架构设计？

或者你负责的一直都是那种公司内部使用的，几十个人用的OA系统，CRM系统？

**团队管理**

你在互联网公司里带过20的团队？

或者你在一个传统IT公司里带过3个人的小组？

就是很多同学平时的工作环境，他没有什么技术挑战，所以只要把系统技术做的简单一些，低成本就可以支撑公司业务了，然后可能你工作了八年十年，技术广度还可以，对流行的技术自己都看过一些书，简单用过，玩过demo。

你的项目经验积累了不少，但是都是一些各个传统领域的系统业务理解较为深刻，没有极高技术挑战的项目经验。

有的人工作时间长，可能就是带过一些人，有过一些带团队的经验，能管人。

### ****追求卓越，自己设立技术挑战****

*要有对技术追求卓越的思维；有对你的团队技术追求卓越的品质。*

自己在公司里，如何通过想办法不断的追求系统的卓越，提高研发效率，支撑也许可能会存在的更高的并发量，更多的数据量，尽可能把系统做的更好一些。

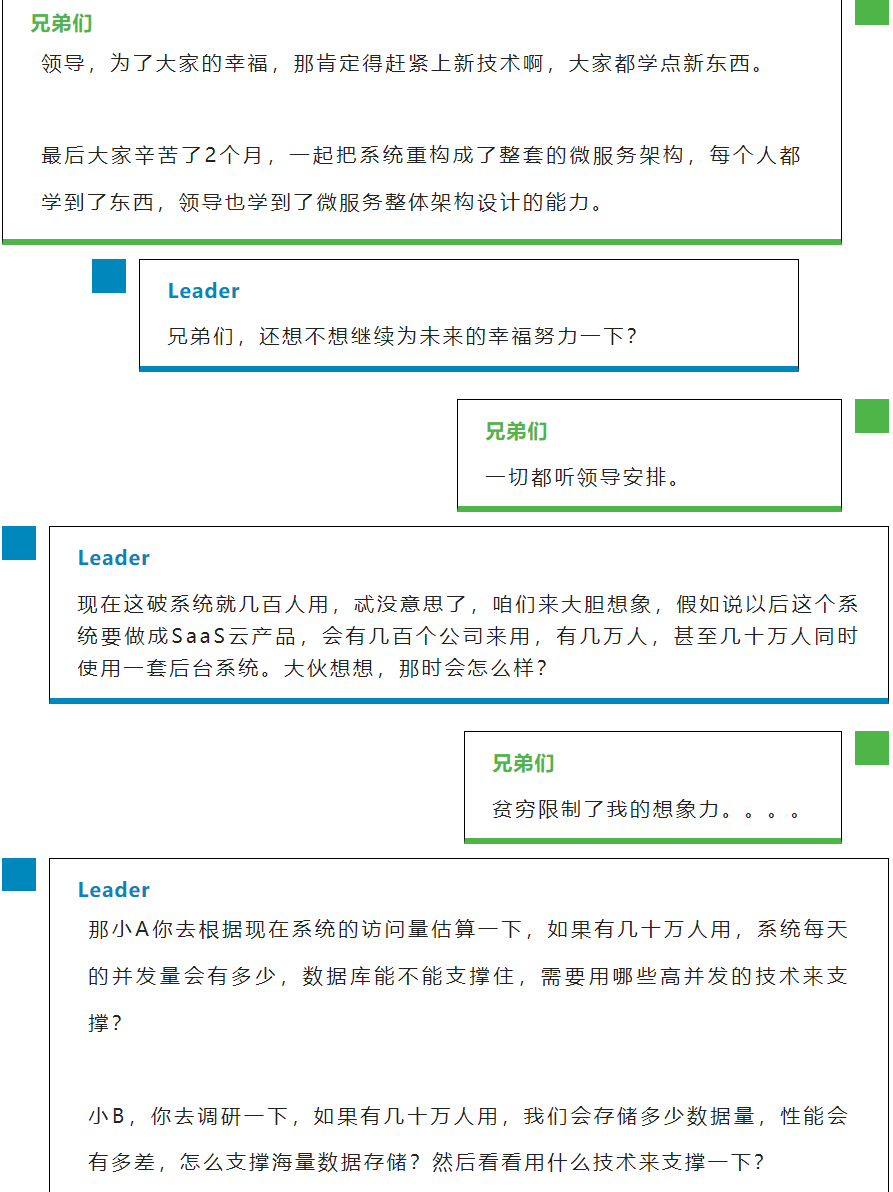
多想想为了解决自己设想的一些技术挑战，如何尽可能把一些业界常用的技术都学习一下，比如缓存、消息、分布式、微服务、大数据，等等。然后都尽可能进行相关的实践，积累相关的项目经验。

例子：

当前：单体，直接Spring MVC + Spring + MyBatis就搞定

主动规划和挑战：





几个月后，大家研发了一套系统，完成了测试，系统集成了缓存集群、MQ集群、分库分表技术，还有很多其他的一些东西。这个时候领导就想办法了，能不能跟老板建议一下，咱们就把产品做成SaaS云的模式呢？然后是不是可以就把这套系统给部署到生产环境里去？

### ****不断提升自己，最后进入BAT****

一旦你开始做到跳槽进入**一家比之前更好的公司**，有更高的技术挑战，那么公司本身的技术挑战就会促使你快速成长，还是举个例子吧。

比如说你本来就在做传统软件的开发，用的都是单块系统涉及的一些技术，就是简单的spring mvc、spring、mybatis等技术做CRUD的业务开发。

通过追求卓越，自己业余不停的学习技术，然后对负责的一些模块多设立了一些技术挑战，自己构思了很多更高挑战的场景下，可以给自己模块加入哪些更高阶的技术。

---

接着带着自己学习的一些技术，还有积累实践经验和思考，进入了**一家创业型互联网公司**。

这家公司的好处在于，互联网公司技术氛围更好，比如zookeeper、redis、rocketmq、sharding-jdbc，等各种技术，在公司生产环境的系统里，都有落地和使用。

那么你此时可以开始真正做一些有点技术挑战的工作了。

---

但是，还是需要继续不停的学习技术，学习更多架构上需要的技术，深入的学习技术，同时积累实践经验。然后带着这份工作经历，同时加上你不断加强的技术学习，你进入了一家比如30亿美金估值的**独角兽公司。**

这家公司有2000万用户，日活用户达到百万级，高峰并发量可以过万，每天数据库里日增数据量达到了数十万。

此时你开始真正接触一些所谓的：高并发、高可用、高性能、海量数据的实际处理。

基于你开发的业务系统，你开始更多的实践，同时你还对各种涉及到的技术有了更加深入的研究，比如对一些核心中间件系统进行了源码级别的阅读和研究。

---

最后你终于等到一个机会，BAT里某家公司让你去面试，经历了四五轮面试之后，对方给了你一个offer，是年薪40万的高级Java工程师的职位。

然后你进去之后，可以在最顶尖的互联网公司里学习开发流程、规范、架构，接触到最大规模的用户量，每天都有解决不完的技术挑战，在这个过程中，你又可以继续成长。

---

最后可能你再次跳槽，就可以进入TMD中某一家，拿下技术专家的offer，在大公司里拿下技术专家的职位，带一个团队，达到人生第一个巅峰。

---

接着你再跳槽，可能一些创业公司就开始挖你去做一些技术管理层。

### ****千里之行，始于足下****

***这里面最难的就是开始的那一步，大量的人都停留在一些完全没太多技术含量的技术工作的情况下，****其实只要能把第一步走好，自己拼命的积累技术，努力跟其他工程师竞争，技术远超跟自己同情况的其他工程师，那么你就有机会率先脱离这种困境，开始慢慢第二步，第三步。*

*到了后面，就是让公司的技术挑战push你不断努力和进步，最后进入BAT这类一线互联网公司。*

## 【offer收割机必备】我简历上的Java项目都好low，怎么办？

### ****高级工程师必备：系统设计能力****

如果你仅仅是对各种各样的技术都熟悉，有技术广度，也有一定的技术深度，实际上是不够的。

如果你的系统设计能力不到位，可能导致你在开发系统的时候会乱用技术。比如有的场景，合适的方案缓存+数据库；可能有的会设计引入Elastic-Search。

互联网行业的面试，很多大厂都会越来越开始问一些系统设计类的问题，举几个例子来看看：

* 如果让你设计一个秒杀系统，你会如何设计？
* 如果让你来设计一个消息中间件，你会从哪些方面来考虑？核心的架构以及数据结构如何设计？
* 如果让你来负责一个电商双11大促系统，你会如何来考虑和设计？
* 我们公司有这样的一个业务场景，XXXX，我给你画个图，YYYY，就根据这样的一个场景以及面临的问题。如果让你来设计这个系统，你会如何考虑？

**面试中的系统设计问题主要分为两类，一类是常规性的，比如秒杀系统的设计，另外一类是那个公司自己的业务场景下的系统设计。**

**第一类系统设计问题**

其实很好快速突击准备，你可以到网上搜一搜互联网公司常问的一些系统设计问题。

收集好一些典型问题之后，百度一些技术博客给出的技术架构设计的思路，将这些思路自己进行整理总结，然后转化为自己的语言，最后落地到纸上画出架构图。

到面试现场，你能够画图把这个系统设计思路说清楚，这个基本就OK了。

突击的话可能几天时间也足够了。

**第二类系统设计问题：**

完全考验的是你实打实的系统设计能力，短时间内针对一个业务场景和技术难点，能否迅速给出一个初步的架构设计和技术方案。

从长期积累和准备的角度，我的建议是在平时自己在开发系统的时候多思考，自己的这个系统有没有什么技术难题，针对这个技术难题应该用什么什么技术，什么方案来解决，这就是潜移默化的在积累系统设计能力。

**对于你要面试的公司，尽可能的多做了解。比如公司业务背景，大概用户量，这些都可以了解一下。**

**如果目标公司有官方的社区或者官方公众号，关注一下，看看上面的技术分享。**

**然后如果面试时，面试官针对他们公司的具体业务出了一个系统设计题目，你如果没有思路，建议尽可能的结合第一类的常规系统设计题目来思考套用。**

### ****如何让你的项目经验更有技术含量****

任何一个面试，都一定会涉及到你作为一个工程师最最核心的价值和能力，就是你做过的项目。

比如一个非常典型的项目经验的面试考察情景如下：面试官反复的追问项目的各个地方的技术实现细节，就想看看有没有哪个地方是有一定的技术难度的，可以体现出这个候选人的一些项目上的亮点。

但是呢，候选人说来说去，总是从业务的角度去说，就说有哪些子系统组成，分别是干什么的，如何交互的，看来看去都是系统业务的东西，就是没看到什么有技术含量的东西在项目里体现出来了。

因为你的项目里没看出来什么东西，没什么亮眼的地方。你看起来就跟千千万万个普通的工程师没任何区别。

**分为两条路线来谈：一个是长期准备型；一个是短期突击，临阵磨枪。**

**平时你工作的时候，一定多给自己设立技术挑战**。总结起来一句话：**没有困难，制造困难也要上。**

**突击准备**，你肯定没有大把时间来付诸实践，但是你一定要自己思考，同时百度一下国内大型互联网公司的技术架构，他们使用了哪些高大上的技术，对于某个技术难点采用了什么技术方案。

## Java工程师如何在1个月内做好面试准备？

### 技术广度

一个合格的适合互联网公司的工程师在技术广度上，还需要掌握其他很多东西。

比如分布式事务、微服务架构、分布式锁、性能优化、Java虚拟机优化、Web Server优化、数据库优化，等等。

可以自己把技术广度可能涉及到的各种点都罗列出来，然后上网百度一下各个技术点常见的一些面试问题，或者是基本的一些核心原理。

### 技术深度

**决定你未来技术的高度有多高，还有你能否成为团队里顶梁柱的，是你的技术深度。**

技术深度，顾名思义，就是说你对一个技术掌握的深度，举个最简单的例子。假如你们公司的一个系统里用到了kafka这个技术，好那么现在问题来了，线上生产环境中，每当高峰到来的时候，不知道为啥写kafka总会偶尔抛个异常出来，或者是说写入到kafka中的数据不知道为啥每天统计下来总会少一些。

**技术深度的快速准备**

如果你在过去几年，自己就深入阅读过大量开源技术的源码，在阅读源码的过程中，自己还把源码的架构图、流程图，手工画了大量的图出来。**此时可以把过去几年源码研究笔记翻出来，快速过一下，练习如何在面试现场在白板上纯手工把各种技术的源码架构图和流程图现场画出来。**

或者在过去几年，你对某个技术领域，针对很高的技术难度做过什么复杂的系统，在那个技术领域扎下了很深的研究的话，那么此时你也可以针对过去的笔记快速的复习，在面试现场一定要现场画图把你做过的复杂系统以及技术难度都说清楚。

问题：你对哪个技术比较熟悉一些？

可以从简历中挑选出某个你相对来说用的比较多，熟悉一点点的技术，然后直接用万能的百度，搜索“XX技术源码分析”，“XX技术底层原理”。通过几天的时间快速的 “ **死记硬背 + 软磨硬泡**”，力争对某个技术相对有一些稍微底层一些的理解。

在打通你的底层技术任督二脉之后，再去对常见的开源技术进行深入的源码研究，比如说：dubbo、zookeeper、spring cloud、redis、rocketmq、elasticsearch，等等。

基础功底：算法

# 常见面试题总结

### 互联网大厂Java面试题：使用无界队列的线程池会导致内存飙升吗？

考点：线程池的运行原理，ThreadPoolExecutor的构造参数有哪些