# 【跳槽篇】

一个合格的Java程序员必然是经历过不同的项目游走在行业技术的前线，有时候除了自己学习他人教导之外，更需要环境的推动，天时地利人和，Java程序员才能在职业道路上走得更远。

## 第一：工作的本质

工作最重要的目的是积累金钱。挣钱这个事情，至少有四种形式：

### 帮别人挣钱（打工）

### 为自己挣钱（ SOHO)

### 雇别人挣钱（企业主）

### 让钱自己挣钱（投资）

第一种门槛和风险最低。而求职只是我们在自己资源和能力不足以进入后几种挣钱形式时，通过出卖自己的劳动能力换取资源的阶段性行为。

## 第二：关于跳槽

跳槽，对IT人来说绝对不是个陌生的词汇，几乎所有人都身体力行过。首先我要强调一点，跳槽并没有坏处，但是频繁的跳槽就不好了。

在工作的前4年，必须跳槽，而且鼓励跳槽(建议经历2---3家公司);每个公司对技术的侧重点都不同，对开发流程控制都不一样，所需要的技术性人才自然不一样;比如外包公司，要的是一个代码狂人(我很喜欢招做过外包的人，感觉很多在外包公司待过的人做事的效率很高);小事业的公司，要的是一个全能型的程序员(能设计分析，能写代码，能部署);大的公司，要的是一个技术扎实，有潜力的人(英语好，理解能力好，Java基础扎实);这样如果你在一家公司呆久了，很可能就模式化，如果一旦离开这家公司，找工作有时候很容易碰壁;多经历两家公司，对程序员很有益!而且我自己也有感觉，每次找工作都在强制自己学东西。

而且关键是，一次跳槽，薪资就更高了 ----- 很多人跳槽都是对当前薪资不满呀。

## 第三：‍跳槽的意义

跳槽其实是打工这种挣钱形态下，我们进行自我调节的方式，通过合理地跳槽，我们可以寻找到市场需求和自我实现的最佳匹配，从而在金钱和成长上双丰收。“市场需求“描述了企业渴求的员工，”自我实现“描述了你想要的生活；当它们重合到一起，梦想就照进了现实。

## 第四：跳槽的原则

永远不要因为”现在很差“而跳槽，要因为”未来更好“而跳槽。只有这样才能保证你一直往上走。

## 第五：什么时间跳槽最合适？

1.认为自己是公司最好的程序员，在公司里已经没有可学习的对象了。

在这一点上，很多人可能是太过自信了，一定要正确对待自己的真正实力，不要太高估自己。如果真如你所想，你真的是最厉害的，那你可以考虑了。还有一种就是，知道自己不是最好的，公司有很多人可以学习，那如果他们都只会个忙个的，没有人愿意将自己的知识和你分享，没有人愿意帮你，那么这时你也可以另有打算。

2.如果你没有学习新的技能，还是用那么老掉牙的技术，你将会被淘汰。

如果你依然用那些过时的技能来开发，那么你将会面临被这个市场淘汰。也许有哪家工作仍在用早期的语言版本，这样你还能工继续工作，但是，你必须明白，如果处在一个停滞不前的技术环境里远比你就职一家不断改进更新的公司里要糟糕的多。

3.如果在这个公司里一直没有做出成绩

倘若你在一个公司里待了很久都没有一个成功的项目，当然这有时也不一定是你的错，原因有很多，可能是公司一直拖延发布，又或是其他的。如果你在接下来的时间里还是看不到前景，那么，这时你可以考虑要不要还待在这个公司了。

4.比起同行，你的薪水少很多又没有什么晋升空间。

除非是有低于市场薪水的原因，例如，公司有很多优秀且愿意帮助你的人，在其中能学习到很多技能这当是最大收获，如果是薪水不多又没有别的可以弥补的，那么你是时候可以考虑要不要从新选择下了。

5.如果公司总是把你当空气，没有让你表现和晋升的机会。

如果是这样，那么，公司根本没把你放在眼里，你也根本不重要，对他们是个可有可无的人。他们觉得不值得在你身体做长期投资，那你现在就可以去找你的新工作了。

6.发现自己在公司待了很长时间，无论技能还是物资都没有太大变化。

每天都是没啥变化的工作，总是只负责一些琐碎的小事，又或是真的有大项目的参与也只是让你负责无关紧要的部分，这样长期下去会让你没有了斗志，变的意志消沉。而薪水方面长期以来也是不尽人意，那么，你还有什么呆下去的理由。

7.现在的你已经没有了当初的工作激情

现在的软件开发并不像有些人想象的有趣，如果你目前的工作做的好无意义，这个时候，如果你有机会挑战一项新的项目才能让你的斗志更活跃，那么，你不妨改变一下现状，可以到能激起你斗志的公司去试试。

## 第六：跳槽的节奏和路线图

比如：最终目标是CTO，那么它的路线图可能是：

应届生—-→初级程序员—-→小组主管—-→部门经理—-→总监—-→CTO。

然后我们会给每一个节点设置一个达标能力和一个参考时间。当我们的能力已经开始可以胜任下一个节点的工作时，我们就会开始寻找相关机会，最理想的是在当前公司进入下一个节点，如果当前公司提供不了的话，那么就要通过换公司来前进。

当然路线图是可以调整的，比如：

应届生—-→初级程序员—-→小组主管—-→初创公司CTO—-→B轮公司CTO—-→上市公司CTO。

但不管怎么调整，自己头脑里边要有清晰的目标。我们总是先在风险最小的路径（比如当前公司）寻求成长不成功时再考虑其他的路径。

## 第七：造成面试不通过的核心原因七大原因

1.能力对职位来说过高或太低

因为能力低而没有机会这很正常，那么能力太高也会影响录用吗？其实这要从稳定性及用人部门领导的需求来看。一般来讲能力很高的人对职位的期望值也就越高，一旦这个职位所能带给他的不管是待遇或是职业成长都比较少，那么不久后就跳槽几乎是完全可以预见的结果，长远的稳定性上来看就比较差了。而用人部门领导对所聘岗位的需求也许就是不需要能力太高，既方便管理也能更好地培养，从事HR工作多年的前程无忧网友“悠然星月”就曾提到：“也许，对方的部门负责人并不热衷于找一个人分担自己的职责，他只需要一个助手，所以，你越是完美，越是通不过部门的审核。”

2.过往经历表现出的倾向不符合公司预期

简单来说，就是以往经历中有瑕疵，而这个瑕疵的影响似乎并没有完全消除，很可能影响你下一份工作的状态。如喜欢频繁跳槽，并且现在仍有这个倾向；为了照顾家庭，难以顾及工作；创业失败，状态尚未调整好；工作年限与工作经验含金量不匹配；求职意向过于混乱，对自己的职业生涯缺乏规划等。

3、讲话没有重点

换句话说，就是废话连篇，言之无物。如果你不能简洁扼要的解释问题，那么面试官也会怀疑你工作的时候是不是也会像你说话一样，拖泥带水。所以在面试过程中，你可以问下面试官，是否需要详细的解释。

解释也是一门艺术，关键是确定重点，如果需要的话再深入到细节。当聊到业务的时候，就应该从业务的角度看问题，不要涉及任何技术术语。学会用简洁明了的方式解释问题。如果你能时刻把握主旨，那么这一点也不是问题。

4、说的太少，含糊其辞

尤其是那些开放式的问题，如“请介绍下你自己”或“请讲一下你曾经解决过的复杂问题”。

面试官会通过你对这些技术和非技术问题的回答来评估你的激情。他们也会通过模拟团队氛围和与你的交流互动来判断你的经验和能力。

如果你不能很好地说明你的经验、成就和技能可以给企业带来的价值，那么你的竞争力就不会搞。所以你需要对一些最常见的开放式问答作充分的准备，学会推销自己。

5、不了解基础的技术问题

面试不是技术竞赛，不是看谁答对的问题多，但是有一些“必须知道”的核心Java和Web基础知识，你不能不知。例如，对于Java开发人员:

* 不知道“==”和equals()之间的区别；
* 不知道equals()和hashCode()方法被隐式调用时的约定；
* 不知道OO的概念和设计原则；
* 不能很好地处理多线程；
* 不知道如何在HTTP客户端与服务器端之间保持状态；
* 不能解释曾投入精力过的应用程序的高层体系结构；
* 不知道SQL。

6、写不好简单的代码，解决不了棘手的问题

作为一个开发人员，你应该根据自己的经验水平，来针对给定的问题和情况编写代码。特别是如果碰到一个比较棘手的问题，那么即使你还没有解决方案，也应该将你的思路讲给面试官听。

当然这在面试时会让人特别紧张，尤其是在还有时间限制的情况下，但是你也必须保持冷静，至少应该说明你将如何试着去解决问题的方法。

7、糟糕的礼仪和态度

迟到，不适宜的着装，抖手抖脚，没有眼神接触，过于紧张，没有提问，显示不出对这份职业的兴趣，“我什么都知道”的高傲态度，贬低你的现在和以前的雇主，遇到技术问题时烦躁不安或者垂头丧气，为自己找理由而不是虚心接受错误，与面试官发生争执，不好的肢体语言，随波逐流而没有自己的看法，过于呆板，撒谎，嗓门太大，无法成为良好的倾听者等等。

## 第八：良好的心态和体态

程序员必须有责任心!这点非常重要。遇上问题，不能随便的踢皮球，很认真的和别人沟通，提出自己的见解，和客户联系，才能把问题解决。技术上的问题不是问题，总可以解决，但是需求上的问题，必须搞清楚!一旦需求出错，一个项目就可能延期或者失败。程序员的责任心决定了他的成长，一个有责任心的人才可能成为骨干，让领导信任。

其次就是能顶住压力，大家都知道程序员要加班，压力大，所有遇到问题必须不轻易放弃，合理的根据能力去处理!

身体是革命的本钱，程序员尤其重要，天天坐在电脑前，很容易发胖，变成鸭梨身材，所有多锻炼呀!

到某些程度就不要跳槽了，在一家有发展前景或者工作开心的工作长久呆下去吧，不能跟年轻人比热情了，这时候要的是稳重，在公司充当的是一个资深的IT从业者，帮人优化程序，分析需求，为项目出谋划策解决疑难等。

温馨提示：多写技术文档

多写技术文档是好事哦-----其实它是对你工作经验的总结和提高

一个愿意和别人分享知识的人，才能在不断的思考中提高，而且写文档很锻炼人滴。

**声明**：整理内容大部分来源于网络，主要是用于学习，分享之用，如有侵权，请联系管理删除，谢谢。