一个菜鸡前端的大厂逆袭攻略(上)

一、关于本人

在正文开始之前，照例还是先介绍一下自己。

本人刚毕业时，从事的是游戏设计师的岗位，后来因为不想再从事游戏行业，裸辞出来用了半年的时间准备考研。喜闻乐见，没考上心仪的学校，就出来找工作。又花了半年时间，在 2018 年 8 月正式入行成为了一个前端开发工程师。最开始的时候因为没有实际工作经验，只能去做外包。 从整体背景上来说，一年职业空白期 + 外包出身，这履历可以说是差得没边了。

两年后不满足于现状的我选择了出来看机会，并且拿到了几乎所有去面试公司的 offer，包括阿里、字节、京东等一线大厂。最终经过一番艰难的思想斗争后，选择了加入阿里。毕竟江湖传言，一个优秀的前端工程师，不是在阿里，就是在去阿里的路上。经过一段时间适应环境后，通过转正答辩，正式成为阿里人的我回头写下这篇攻略，以期帮助到有心进入大厂的小伙伴们。

二、翻盘策略

虽然开场拿了一手烂牌，但是不要慌，稳住我们能赢。接下来复盘我在整个过程中坚持的策略。

三、学习篇

3.1 保持学习习惯，利用好碎片时间

既然从事了程序员这一行，那么持续学习这件事自然会伴随我们的整个职业生涯。技术的变迁和发展速度很快，在我入行的时候，angular.js 依然火热；等到如今这个阶段，除个别老项目以外，你很难再见到有人使用它了。保持学习习惯这件事，是一个技术从业者所必备的。

而在学习过程中，最重要的品质就是坚持。人都是容易懈怠的，如果没有强大的自驱力，三天打鱼两天晒网，很难保持学习的质量，技能的提升也就无从谈起。很多人可能会觉得，平时工作很忙很累，哪来时间学习呢。事实上，时间就像肚子上的赘肉，只要想挤，一定是可以挤出来的。以我自己来说，即使再忙的时候，依然会保证在每天上下班通勤的路途中学习，平时也会利用工作和生活的碎片时间投入到学习中。虽然每天花费的时间不见得很多，但长期坚持下来，也能获得显著的提升。当然，如果能有整段时间投入学习中，就更好不过。

3.2 脚踏实地，相信心急吃不了热豆腐

在几年前，前端这个岗位要学习的内容并不多，会点 html、css、js、jquery 就足以应付工作和面试了。但随着前端技术的不断发展与迭代，现在要从事前端这一行，需要会的东西既广且深，在每一个领域都需要投入大量的时间去细细研究。这时候最容易出现的问题是心急，急于将一个模块快速学完，再投入到下一个模块中。殊不知欲速则不达，越是心急，越是难以静下心来，扎扎实实掌握好。

在过去这两年多的时间里，我从来没有产生过“要赶紧学好抓住机会去大厂”这样的念头。对于我自己来说，学习的意义就是能力的提升，只要还在持续前行，那就是有价值的。所以我在研究任何一个模块的时候都能沉下心来，从它的使用方式到原理逐层深入，不在意这个过程中花费了多少时间和精力。

举个例子，前端工程师都会学的 JS，我一开始也是从基础的原型链、闭包、异步这些机制开始学习了解。后来随着知识和积累的逐步提升，就不仅满足于知道 JS 存在哪些机制，还会去思考为什么会存在这些现象。所以这时候会开始看一部分 V8 的源码，了解 JS 底层运行的机制，从而了解这些现象产生的原因。当你对底层的原理有一定了解，表象上的问题自然迎刃而解。

3.3 高效学习，及时复盘成果

在这个时代，努力学习的人很多，但努力以后能获得好结果的却并不多，这里很重要一个原因是大家的学习效率完全不一样。人的时间有限，如何利用好它至关重要，所以高效的学习方式必不可少。但什么是高效的方式呢？

我之前曾经见有人说，看官方文档是最好的学习方式，看视频的都是学习能力很差的人。我对此持保留意见。个人愚见，针对一个完全陌生的模块，前期的学习重点是如何快速上手、能够学会最基础的应用方式；深层理解和深入原理是后期的目标。学习是一个循序渐进的过程，如果有视频可以快速上手，这就相当于是上学时有老师带着你先勾画重点。所以我比较推荐这样的学习方式，即先通过视频快速入门，再通过官方文档探究细节，最后研究源码来深入原理。

既然学习需要资料，那么如何找资料也是一件很重要的学问。这里需要感谢 B 站及其中许多 UP 主，在我还是个刚入行的萌新之时，提供了大量免费的视频学习资料，让我得以快速掌握一门新技术。到了后面需要深入研究的时候，京程一灯的公开课给了我极大的帮助，为此我在后来也报名了他家的精英班（虽然精英班我到目前都还没结课）。

阶段性的学习完成以后，需要及时复盘、总结，这里推荐我用到的两个方法。一个是面向镜子，假设在给镜子中的人讲解刚学习到的知识点，并在全过程录音，结束后通过回放去听自己的讲解过程是否流畅，逻辑有没有梳理清晰；或者是输出一篇技术文档，总结知识要点，在梳理过程中对知识的理解会更加深刻。我一直相信，衡量有没有学好一个知识点的最佳方式，就是看你能否把不懂它的人给讲懂。

3.4 不去问“这个学了有什么用”

一直以来，我经常会听到身边朋友问这类问题：前端有必要学 docker 吗？AI 学了对前端有什么用？

回顾过去，在我开始决定要从事技术工作以后，就从来没有思考过学习某项技术有没有用。事实上，除了前端该掌握的基本知识以外，对于后台、数据库、运维、AI、图形学、流媒体这些领域，我都主动学了个遍。当时的想法很简单，既然不知道有没有用，那就先学着呗，万一哪天用上了呢。

事后来看，功利点说，在我后来的面试过程里，阿里问了后端和运维技术、字节和京东问了 AI、bigo 全程都在问音视频相关知识。安卓/ios 原生开发和图形学更是贯穿了整个面试流程。因为我对这些领域都有着大量的研究，所以很自如的回答上来了面试官提出的各种奇怪的问题，offer 也就一个个水到渠成收入囊中。

工作中也有使用到的场景。在我入职阿里以后，基于从前所学过的知识，使用 GO 语言配合 JS 改造了一个开源产品，参与了业务中数据库的表结构设计，虽然这些在大多数人看来都不属于前端的职责范畴。

当然，以上都基于一个前提，就是在此之前我已经把前端的基本知识掌握到了一个还不错的水平。不然如果连 react 和 vue 的源码都没弄明白，就去研究 spring，那就是舍本逐末了。

四、工作篇

4.1 主动承担高难度的工作

在过去的两年时间里，身在外包的我几乎每个季度都拿着公司最高的 S 绩效。这一方面是因为我在平时工作中展现出来的能力得到了赏识与认可，另一方面是我在工作中主动承接了很多具备技术挑战的活。人的时间和精力是有限的，如果大量投入到简单的业务逻辑中，将难以得到显著的能力提升。就像很多人说具备五年工作经验，实际上他只是将头一年的工作经验重复了五次，这对你的职业生涯没有太大帮助。

机会是不会从天上掉下来的，它需要你自己去争取。当初在面对一个困难而复杂的需求时，同事们害怕背锅、拒绝麻烦，都不愿接手，此时我主动找领导去申请了这个机会，并在一年以后拿到了令人满意的结果。这个项目促成了两家知名世界 500 强企业的合作，700W 粉丝的总裁为它单独发了一条微博。虽然这中间实现需求的过程确实非常痛苦，各种奇怪的场景完全超越了我最开始的想象。但当我做完这一切回头再看，才发现原来在过程中我解决了那么多问题，并借助研究问题的过程学习了很多新的知识，这对我的能力提升帮助之大，是我最开始万万没有预期到的。引用入职后从阿里 CTO 程立先生那里听来的一句话，学习的最好方式是在工作中遇到困难并尝试去解决困难，因为你会带着问题去学习，所以你的效率比任何时候都高，学得比任何时候都更扎实。

如果你拒绝参与有挑战的工作，那么也是在拒绝让你自己成长的机会。

4.2 发掘项目的亮点

众所周知，找工作的过程中一定会被考察到的一个点是项目。

假想在面试的场景里，面试官问你：你觉得你参与过的这些项目中，解决过最难的问题是什么，或者最让你有成就感的一个挑战是什么。你想了半天，最后表示好像也没做过什么特别有挑战的事情。这样面试官没考察到你解决复杂问题的能力，容易对你产生怀疑，你也失去了一个绝佳的展现自己的机会。所以项目有亮点是一件很重要的事情。

但是大多数时候，我们从事的项目都更偏向于业务侧，画个界面写点 JS，框架里成熟的 UI 组件库套进来，基本就解决 80%的问题了。剩下 20%的所谓复杂问题，最后发现好像也就不过这么回事，上网百度谷歌搜下，一般都有现成的解决方案，甚至代码都给你写好了，复制粘贴就能用。上文提到的高难度工作并不是随时都有，该怎么办呢。

我最开始也有这样的困惑，尤其身在外包，核心的工作内容接触不到，做的活都非常简单且基础。直到后来有一个任务，让我去做关于音视频的解码投屏渲染。其实提需求的人已经提供了一套完整解决方案，我只需要将这部分的代码导入进来，剩下的几乎没什么工作需要再做。但我意识到这是个绝佳的机会，所以立刻去利用工作之余的时间研究了市面上所有能找到的技术方案，甚至包括 C++中的 ffmpeg（此过程中又实践了一波 wasm），在比对学习完以后，使用一个效率更高的解决方式替换了原方案。通过这次实践，领导认可了我的工作表现，以至于后续将更复杂的任务派给了我；而我自身也成功积累了知识储备和吹嘘项目的资本。

如果工作中没得到发掘亮点机会，也有别的途径可走。比如参与开源项目，或者自己开发一个小项目。我选择的是后者，在工作之余抽时间做了关于数据埋点和错误信息收集的 SDK，这也可以成为你项目中的亮点（虽然我在面试的过程中没有用上）。