

技术人员升级打怪的方法论

太白上仙自己在带团队方面有着非常充足的经验，由本上仙直接招到公司的小白 coder 中，在 18 年的时候就出了两个阿里的 P8 啦！

而跟着本上仙做过项目成长为 P8 已经有 7 个人了！唯一可惜的是目前还没有人升到 P9。

本上仙自以为在带小白方面特别有经验，能够迅速挖掘每个人的天赋，让大家在尽可能短的时间内提高自己的实力。

本上仙最近把这一套经验提炼成了一套理论，希望能够对每个技术人员的职业道路上都有一些帮助！

1、核心能力

如果问技术人员最核心的能力是什么？

我相信每个大厂的 Leader 都会做出同样的回答—— **trouble shooting 的能力**。

这种能力是真正能拉开人差距的能力，因为技术不断在创新，没有人能见过所有问题，我们每个技术人员都是在不断学习不断成长的。

但是在招聘和评定职级的时候，我们很难这么快的去评定一个人解决问题的能力到底有多强。

所以我们需要了解每个同学的过往项目，考察他在过往项目中学到的点，从而通过一些能迅速评定的能力，来判断各位同学 trouble shooting 的能力。

2、考核剖析

技术界的老前辈们，总结出有三种公认的可以迅速判定，且不会有太大出入的能力。

即基于项目的基础、业务设计、算法。

所以，到目前为止，公司进行技术评定，不管是面试笔试，一定是从三个方面来考察求职者的技术能力。

根据不同岗位对这三方面的需求差异，企业又把技术岗分为两大类，分别是开发岗位和算法岗位。其中开发岗位的考核点是着重于基础和一个好的业务设计，对算法的要求并不高；

而算法岗位的考核点则是着重于算法，其次是基础，对业务设计方面要求不高。

因为算法是需要一定的天赋，同时也是这三方面中比较难的，所以在同一职级（资深技术专家以下）中，算法岗位的薪资是高于开发岗位的。

在企业的考核中，不管是算法岗位还是开发岗位，求职者想要应聘成功拿到高薪，就要有与相应职级要求的项目经验而匹配的三方面的能力。

3、职级剖析

每一个打工人的动力毫无疑问，就是钱。谈钱不伤感情！

在绝大多数情况下，每个打工人的薪酬就代表了他的技术职级。

本上仙把 100 万以下年薪以下的技术分成了 8 个层级，为了方便大家理解，本上仙都写到黑板上了，如下图所示：



职级	薪酬范围（杭州基准）	职级名称	要求年龄
L1	0~6W	未知	22~24岁（校招）
L2	6W~10W	学徒级开发工程师	22~24岁（校招）
L3	10W~15W	入门级开发工程师	22~24岁（校招）
L4	15W~20W	初级开发工程师	22~24岁（校招）
L5	20W~40W	开发工程师	25~28岁
L6	40W~50W	高级开发工程师	25~28岁
L7	50W~70W	技术专家	26~32岁
L8	70W~100W+	高级技术专家	26~32岁

这个表格基本上 L 就是 P 的意思，也就是比如你处于 L7 这一级，你在阿里系就是 P7。

对于大厂来说，招人都是 L4 以上，一线大厂都是 L5 起。

最右侧的要求年龄是指你要在要求的年龄达到这个水平才好入职大厂，当然对于晚熟的同学来说很伤，不过招聘是个 case by case 的事情，凡事都要结合实际情况来看。

当然，本上仙希望看了这篇文章的同学都早熟，披荆斩棘横推各路 offer ！

4、选定项目

大家根据表格知道自己所处的技术职级之后，如果想要打怪升级，就要仔细研究本上仙说过的这句话了：“基于项目的基础、业务设计、算法。”

每个读者肯定都是聪明绝顶（并没有说大家秃顶），这句话已经讲得很显然了，首先就是根据自己的升级目标选项目。


在选项目这个事上，如果已经工作的，我建议大家的项目基于工作内容去选，就算最简单的工作也能问出很深很深很深的问题！

每一个人的经历不同，他的能力必然有很大的差异，并不是别人会什么你就要会什么，关键要看能不能理解透彻。

对于 L7 及以上的同学，我相信大家这点理解已经很透彻了！

对于还没工作的同学来说，我帮年轻的同学们选了四个渠道，本上仙把这四个渠道按照优劣不等排序分为上上策、上策、中策和下策。

为了方便同学们阅读，我也写到黑板上了！



选项目的渠道

评级	渠道	分析
下策	跟着传统培训机构或网络免费的实战项目做	缺点：项目错误多，高度雷同，完全不考虑个性化。 非常不推荐！
中策	进入不好的公司实习	优点：自己真正在接触项目；缺点：不好的公司各有各的问题，说不定会学到坑，将来成长会暴雷。
上策	进入大厂实习	优点：都是优点；缺点：难进！
上上策	大厂 L8 及以上带着远程实习	优点：都是优点；缺点：费钱啊！

那有没有十全十美的办法呢？未来可能有，比如本上仙在思考能不能让 AI 代替这些大厂的 L8 以上的来教大家，买个软件总比雇个 L8 便宜得多啊！

可惜了，这个方案本上仙还在研究呢！因为照着目前已有的 AI 算法，理论上就基本搞不出来，需要先把算法更进一步才有希望！

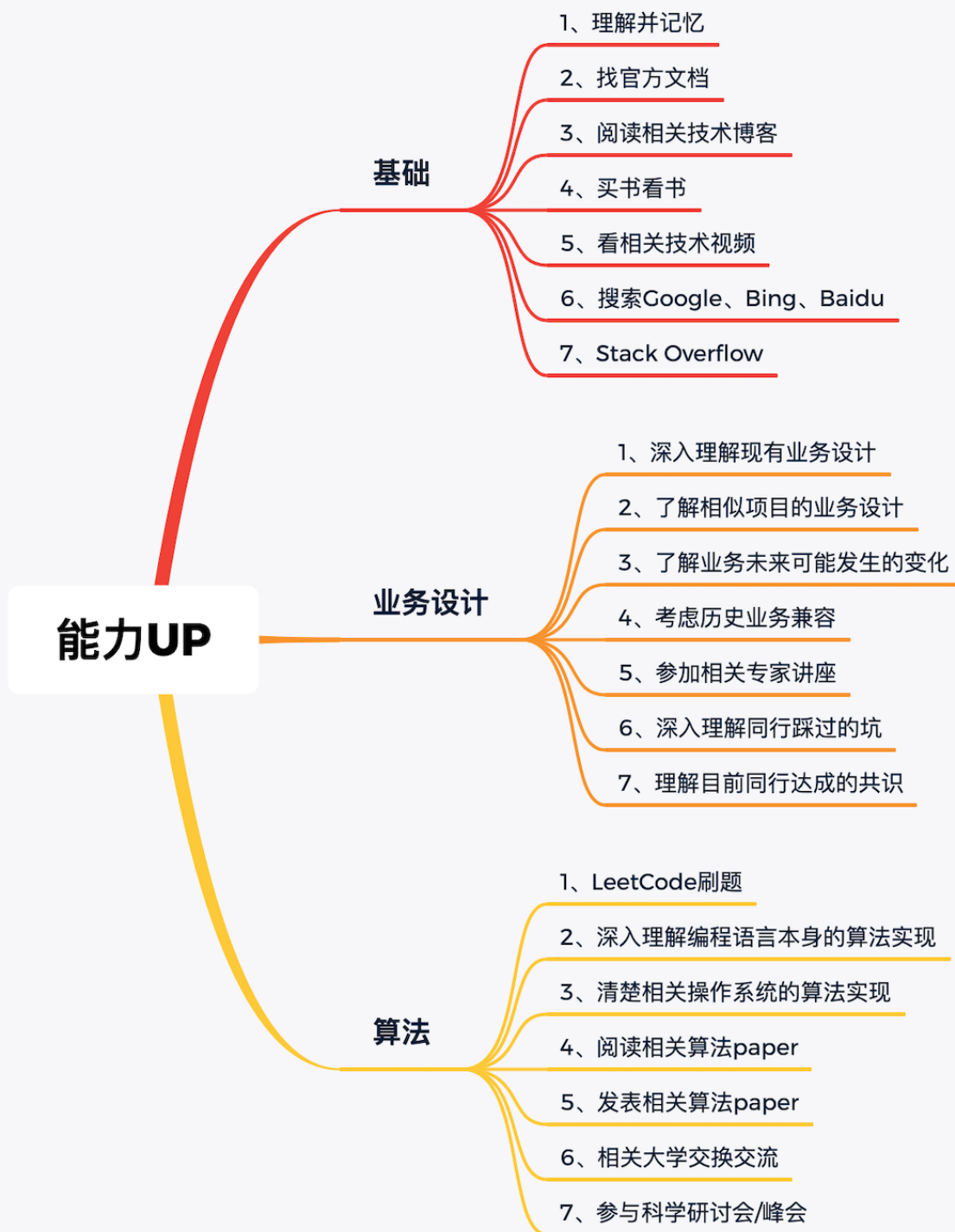
5、能力 UP！

选定了项目之后，如何 up 自己三方面的能力呢？

其本质就是要了解你的这些项目里用到的相关基础、业务设计和算法。

具体的提升办法，我也给同学们画了个脑图！

如果你需要这张脑图，请麻烦关注我的公众号：“太白上仙”，回复：“666”即可。



本上仙可没亏待任何一个级别的读者啊！给每一个能力点都写了7种办法，从0基础的学生到P12的科学家都能找到自己该如何提升哦！

好吧，请P12的诸位科学家轻拍！小仙路过而已。

6、核心理论

好了，在讲了这么多之后，本上仙给大家在上面基础上抽象一下我的这套核心理论——帅气太白打怪升级论！

因为大家都是码农，我也不多说了，直接写了点伪代码，大家一看便知！

```
1 //每职级Li，分为Li-,Li,Li+三个小级别。
2 技术能力 = 分析现有能力；
3 //获得目标职级，当前职级+1
4 i = 技术能力对应的职级脚标 + 1； //比如现在是L1级，那目标职级就是L2。
5 项目 = 基于现有工作/生活经验的项目（必须完全个性化）；
6 //能力 UP 循环
7 do{
8     //1、上来先选定项目，明确要干嘛
9     项目 = 项目 + 选定Li级别所需的项目内容；
10    //2、先预习理解基础
11    学习项目所需的基础；
12    //3、预习完直接实战，码农干的是工科，不是理科！先干了再说！
13    实战项目；
14    //4、总结项目所需的基础、业务设计和算法三方面能力
15    总结相应基础、业务设计和算法三方面能力；
16    补给相应基础、业务设计和算法三方面能力；
17    //5、继续分析一下学完之后的能力
18    技术能力 = 分析现有能力；
19    //6、没达标就反复训练
20    while(技术能力 < Li+){
21        //7、请教高手
22        高手指点项目相关
23        //其他三方面还是靠自己
24        补给相应基础、业务设计和算法三方面能力；
25        //继续分析一下学完之后的能力
26        技术能力 = 分析现有能力；
27    }
28    //8、了解现有项目下个职级级别的方向
29    了解下个职级级别的方向；
30    技术能力 = 分析现有能力； //再分析一波，可能会出现跳级
31    i++; // 加一下次循环
32 }while(i <= 8 && 技术能力 >= L[i-1]+);
```

对于带团队来说，这套办法屡试不爽！绝对能够快速帮助自己团队的队员成长！

比起纯粹靠天靠脸吃饭，这套理论能让你至少学习进步的时候有个方向！

而不是无头苍蝇一样逮住什么学什么，然后学了很快就忘记了，干了几年还升不了职级拿不到高薪 offer ！

7、理论举例

毕竟理论太抽象，我们举个简单例子来实际看看。

当然，完整的例子是很长的，小仙功力有限，没办法写上几万字，所以只能挑一些重点来讲。

比如对于 Java 初学者来说，其入职目标是 L2 ~ L6 级，但是学要从 L2 开始。

你一开始不要上来就学什么多线程、RPC、RMI 这种，不经过任何实战去一直学习，只能把自己立刻搞晕。

这种不实战一直学习的办法基本上是自己劝退自己！

如果想往算法方向发展的，可以在学了基础之后，立刻去 LeetCode 刷上一些题，至少你就有地方去写代码了！计算机是个工科，你只有 coding 才能提高自己。

如果算法没天赋，想要做项目，就要先去理解接口和数据库设计，先去把基本的 SQL 语句搞明白，搞个项目跑跑看。把你的项目搞通了，搞明白了觉得能达标 L3 了再去看多线程不迟！

其实很多初学者对技术的理解有极大的偏差，觉得用个 Runnable，new 个 Thread 就是会线程了，在 SQL 语句都没搞明白的情况下去看这个，最后只能是自己一团糟！

对于技术的学习我只想说四个字——**先深后宽**。

深度不够，你就戳不中面试官的 G 点，面试官就觉得你很无聊，你就拿不了好的 offer ！

关于深度的把握，我就拿存一个文件这种简单的业务来说，对于 L2 应该明白文件写入用哪个类哪个方法；对于 L3 就应该明白 IO 相关类的区别；

L4 你要对文件系统有个基本的概念；L5 需要知道文件系统和分布式 KV、分布式文件系统、数据库优劣的选择；

L6 至少能够对应用分布式 KV 可能出现的问题有所了解；而 L7 需要明白如何设计一个文件系统包括其功能、格式，计算机是如何组织分配、保护和检索文件的。

8、实践难点

这套理论是完美无缺的么？当然不是！

这个难就难在选项目是个很难的点，要针对每个人选出正确的项目，他才能迅速成长。

单纯的基础性学习是毫无意义的，因为我们希望团队里的成员靠着这套方法论最终获得的是**trouble shooting** 的能力，

在使用这套理论的时候，需要特别注意，一定要在选了合适的项目之后自己动手去做起来，这个就是锻炼你**trouble shooting** 能力的时候了。

这个就好比你去工地挖个坑（我们都是码农么，拿工地打比方很恰当，哈哈哈），光看是没有用的。

因为有的人肯定说你要先理解每粒砂石的摩擦系数，然后通过微积分计算铲土的力道；也有人说需要先学会铁钴镍在不同温度下的化学分子特性，从而更好地使用铁锹这一工具。

众说纷纭，不如自己动手铲两下！铲了你才知道你需要的是什么，否则在这个网络信息爆炸的年代，说什么的人都有。

而本上仙的这套方法论，是希望同学们通过项目锻炼**trouble shooting** 的能力，从而获得相关基础、业务设计、算法的能力。

如果喜欢太白上仙，可以关注 **【太白上仙】** 公众号
也可以关注太白上仙的[github](#)