链滴

同样都是开发，为什么你不如别人？

作者：xuexiangjys
原文链接：https://ld246.com/article/1589738613510
来源网站：链滴

2020年由于疫情的影响，大批量的公司破产倒闭，即使能坚持下来的，也是推出了很多财务削减和人裁减计划（也有美名为人员优化），这导致了大量人员的失业，当让也包括了我们这些做开发的程序。

疫情时间，为了能快速找到工作，很多人又开始四处寻找面试材料复习开始备战面试，但就在复习的程中有些人可能会发现，原来自己工作了这么多年，水平可能都不及一个拥有三年开发经验的新人。

那么问题来了，同样是开发，为什么你不如别人？如何才能让自己变得更加优秀？下面我将从三个方阐述我的思考。

1.做事的艺术

在工作中，我们可能会碰到各种各样的问题，如何优雅地处理这些事情，非常考验一个人的能力。

严谨和一丝不苟的态度

有人经常这样向我抱怨：面试造火箭大炮，工作拧螺丝。我想即便你是拧螺丝的工作，也务必要保持丝不苟的态度把这拧螺丝的工作做好，否则你的这步没能拧好，很有可能导致造出来的整个火箭还没上天就爆炸了。

我觉得干咱们研发这一行，严谨和一丝不苟的态度是必须具备的基本职业素养。因为可能就是你的一小疏忽或者情况考虑不周，可能给别人带来多大的麻烦和后果。

可能一、两次地这样坑别人，别人还能原谅你，帮你填坑。但如果由于你的不严谨和不认真的态度，致三番五次地坑别人，时间长了即便你人再好，别人也不会再相信你，这样你在团队中就举步维艰了。

把握问题的本质

在工作的时候，我们经常会遇到各种各样的奇怪的bug。面对这些bug，不同的人处理的方式也是不相同。有的人是恨得咬牙切齿，恨不得和提bug的人干一架；有的人则是非常淡定，一边询问bug的细情况，一边静静地坐着打断点、打日志分析问题。两者处理的方式不同，带来的结果也不尽相同。

那么当我们在开发过程中遇到问题时，我们该如何解决呢？我想核心的解决方法就是 把握问题的本质。

如何 把握问题的本质，以下是我常用的方法论供大家参考：

1. 了解问题的详细情况，收集问题出现的条件和现象：只有了解问题才能解决问题。
2. 模拟问题出现的场景，对问题进行场景复现。
3. 将偶现问题转化为必现问题，从中寻找规律。
4. 善用排除法，筛除干扰项。
5. 断点+日志相结合进行问题跟踪，深入源码探寻问题的本质。

一旦把握了问题的本质之后，一切便会迎刃而解。后面你需要做的就是找到方法，并解决它。你可以己想办法；搜索网上有没有人和你遇到同样的问题；请教这方面熟悉的人...

事前永远比事后更重要

事前埋下的坑，也许需要事后付出数倍的努力才能把坑填完。很多时候我们经常会忽视事前计划、设的重要性，往往是走一步看一步，等功能实现到一半的时候才突然发现这条路越走越崎岖或者根本行通，这个时候你是非常难受的：继续走下去，可能前面的坑会越来越多；不继续走，重新想方案，项

延期，进度赶不上，要被问责。

因此在做任何事前之前，一定要自己要做的事情想清楚了再去做，避免南辕北辙的尴尬。

以下是我给出的几点建议：

1. 在做一些较为复杂的功能前，尽量做好设计。这里的设计主要包括：

- 流程图：把所有可能出现的情况都考虑进去，越详细越好。
- 设计类图：包括UML图和时序图等。好的设计望望事半功倍，这里我推荐大家多学学 设计模式
- 性能设计和可拓展设计。

2. 养成良好的编码规范，在关键的、难懂的地方多加些注释，这样可以避免长时间后的遗忘，导致代晦涩难懂，大大增加维护难道和bug产生的几率。

3. 提高代码的质量，在实现功能的同时，注重代码的性能，对于一些常见的性能问题要烂熟于心。

4. 在问题出现任何端倪之前就立马进行解决，即使不能完全解决也要预先想出替代方案。否则时间长或者上线了之后，你可能需要付出数倍的精力才能解决，并有可能带来非常不好的影响。

低调做人高调做事

Talk is cheap. Show me the code.这句话可谓是IT圈里最朗朗上口的一句话。

咱们干研发这一行的不同于其他职业，并不需要极力向外推销自己来获取更高的业绩。我们绝大多数研发人员都是务实派，靠的是一行一行码出来的代码去实现自己的价值，少说话多敲几行代码会更有值得多。

所以，那些成天夸夸其谈，开口就是讲上一堆技术架构，闭口写起代码又是一团乱麻的人，是比较不欢迎的。

我们做技术的不要成天拿着技术出来显摆。要知道人外有人天外有天，比你技术牛逼的大有人在，没要整天要在技术上比个高低贵贱的，也不需要刻意让别人知道自己有多么厉害，因为你写的代码就能明你的技术水平，时间一长大家自然心知肚明。

帮助别人的艺术

在帮助别人的同时，还能让自己对这块的技术掌握得更加透彻，何乐而不为呢？

帮助别人，而不是施舍，这一点尤为重要。我们要乐于助人，但是也要注重方法。帮助他人是建立在互尊重的基础上的，否则你的好心帮助会被别人理解为同情施舍或者多管闲事。

因此我们在帮助别人的时候要注意以下几点：

- 不要有帮助人的企图，只有在别人需要帮助的时候才去伸出援助之手。
- 给予被帮助人最起码的的尊重。
- 不要借着帮助他人的名义去干预被帮助人的成长，最好的帮助就是点到即止，剩下的顺其自然。

2.学习的艺术

从事开发工作，无论你是在产品线上写业务代码，还是在技术平台进行技术研究，我们都不能放弃学，放弃对新技术的尝试。放弃学习就好比战士上战场弄丢了自己的枪，很快你将会被一浪又一浪的技浪潮所淘汰。

学习和汲取他人的长处

对于大多数的人来说，发现别人的缺点很容易，但是发现别人的优点却很难，这也是很多人不能快速长的原因所在。

优秀的人总是善于发现别人的优点并加以学习。学习、模仿并最终超越是他们战无不胜的秘诀。他们不在于你身上有多少缺点，他们只在乎能从你身上学到多少东西。

他们不仅会向身边的人学习，还会向以下几个方面进行学习：

- 优秀的源码。这里包括系统源码和优秀的开源项目源码。
- 优秀的技术书籍、文章。
- 优秀的理念和思想。

把握学习的广度和深度

漫无目的地学习必然导致效率的极其低下，我们在学习之前一定要给自己设定一个目标：到底是想学不同领域额的技能，成为一名全才；还是想就某一领域深入研究，成为一名专才，这就涉及到学习的度和深度的选择了。

因为你不同的选择可能导致不同的人生轨迹，就目前而言，大厂更偏向于某一领域的专才，而小厂更向于拥有更多技能的全才，当然这也不是那么绝对。就选择而言，大厂固然很好，但是又有多少人能入大厂呢；小厂虽说待遇各方面都不敌大厂，但是小厂的机遇多啊，说不定哪天公司发展得不错，你能爬上领导的位置了呢。

所以无论你是选择学习的广度还是学习的深度，其实都是没有错的，唯一错的就是你压根就没有思考这事。

当然这里的选择也不是绝对的，每个人在不同的阶段可能选择的方向并不相同。当你初涉社会刚开始作，你可能追求的是学习的广度，但慢慢的当你对某一领域感兴趣或者表现出异于常人的天赋时，你能又会转而追求学习的深度。

每个人的技术都有可能在某一刻达到瓶颈。如果现在的你翻开一年前你提交的代码，却发现和你现在交的代码并无差别时，这个时候你就要小心了，很可能你已经达到你的技术瓶颈了，这个时候考虑换个学习的纬度可能是不错的选择。

持续不断地学习

技术在日新月异地不断变化和发展着，前几年还比较流行的技术，可能没过几年就被人们所抛弃。当命性的突破技术取代旧的技术时，这是历史巨轮不断向前发展、不可阻挡的趋势。

不要以为你现在掌握的技术就能够养活你一辈子，我们需要对技术的发展保持着极大的敏锐触觉。一你所掌握的技术逐渐被新技术所替代时，你就要小心了，可能留给你学习的时间不多了。

利用好学习的工具

利用好学习的工具，能够极大地提高我们的学习效率。

这里我主要介绍关于自学一门技术可以利用的工具：

● 专业性的入门书籍。对于新手和小白而言，我还是建议大家先找几本专业性和权威性最强的书籍作自己的入门指南。因为书本讲解得更详细也更成体系，对于入门而言还是相当不错的。

● 专业领域较强的技术论坛和博客。在这里我们可以学到很多书本上所没有的一些前沿技术的资讯和术交流心得。这里推荐掘金和思否。

● 开源代码托管平台。在上面拥有海量开源的项目，其中也不乏许多优秀的开源项目可以供我们学习参考。学习和借鉴别人优秀的代码和设计思想，能让我们快速提高自我的coding能力。这里我主要推荐ithub和Gitee。关于如何使用开源代码托管平台，可以参考我之前写的一篇文章：《你真的会使用github吗？》。

3.提问的艺术

我们每个人都不是万能的，都会遇到很多我们不懂的问题，需要向别人进行求助。但是并不是每个人问题都能够得到别人的答复，这完全取决于提问者提问的水平。

这里我先模拟一个场景：当你在github上使用了某人开源的轮子时，遇到了问题需要向作者提问或者issue，你会怎样进行提问？

● 提问者A：大佬帮帮忙，我在使用xxx的使用遇到问题了，请问怎么解决啊？

● 提问者B：老哥，我说xxx小白，在使用你的xxxx搞了一天了都没有运行得起来，能不能帮帮忙，我在没办法了。

● 提问者C：您好，大佬。我在使用xxxx时，频繁xxxx，导致xxxx,但是xxxx, 又会xxxxx, 但是呢...(以省略500字)。可能我形容地不是很清楚，你可以试一下就知道了。

● 提问者D：请问怎么解决，.....（以下省略数百行的日志）

● 提问者E：你这xxx根本不能用，.....（以下省略约一百字的抱怨的话）

● 提问者F：您好，我在使用xxx的xxx版本的时候，遇到了xxx问题。下面是我出现问题时的现象（....）及日志(....)。我是这样xxx,然后xxx,最后导致xxxx。我出现问题的设备型号是xxxx，在xxxx上没有出现题，是不是xxxxx导致的？

上面6个人提问的方式是完全不同，我想可能只有提问者F才能够最终得到别人的答复并顺利解决问题下面我将帮你逐一分析原因：

● 提问者A是很多人经常犯的错误，那就是只抛出了问题，并没有给出问题出现的现象和依据，这会被提问者无从下手，没有丝毫回复的欲望。毕竟你是求别人帮你解决问题，而不是领导发号施令。

● 提问者B是很多初学者（学生）常犯的毛病，没有明确的问题，没有明确的解决预期，有的就是祈式的求助，甚至连要帮什么忙都没有表述清楚。对于这种提问，绝大多数人是不予理睬的，因为他们不想把时间浪费在一个什么都不明白的菜鸟身上，毕竟他们不是你的老师，没有义务教你基础知识。

● 提问者C的问题就是说得太多，没能精确描述问题是什么。这样表述不清的提问只会让被提问者满的问号，然后直接回复：？？？。

● 提问者D也是很多人经常犯的错误。出现问题之后的第一反应不是先去进行一番思考尝试自己解决而是无脑地将一堆无用的错误日志贴出来，请求别人给出解决方案。对于这样的问题，可能绝大多数的第一反应就是：能百度解决问题的，请不要来烦我，谢谢！

● 提问者E就不用多说了，这种提问明显不是冲着解决问题的目的来的。对于这种不友善，怀有敌意提问，我想大部分人的反应不是去帮忙解决问题，而是在想：这人不会是傻*吧？

分析了上面人的提问方式后，我们可以总结出如下几个问题时的技巧：

- 首先明确问题是什么。
- 优先自我思考解决，解决不了再向别人寻求帮助。
- 清晰地描述问题。
- 问题解决及时反馈并表达谢意。

明确问题是什么

在提问之前，首要任务是要搞明白自己到底要问什么，诉求是什么，这是对被提问者最起码的尊重。

什么都没搞明白就稀里糊涂地跑去问别人，这会让别人觉得你很唐突无知，给人留下非常不好的印象这也会直接导致别人不愿意帮助你解决问题。

为什么这么说？因为别人要想帮你解决问题，还得先搞明白你的问题到底是什么，你的诉求是什么，后还需要帮你分析问题出现的原因，最后才能帮你想出解决的办法，这花费的精力和代价实在是太大。要知道这不是在学校或者医院，没人会愿意这么大废周折地帮你。

所以，要想自己的问题能够得到别人的答复和帮助，你必须想明白你要问的问题到底是什么！

优先自我思考解决

你可能也会遇到这样的情况，经常有人会在论坛上、qq群里、博客评论区里，动不动就贴出数百行的错日志，然后不加一点说明，开口就问：这是什么问题，能帮我解决吗？

像这样不经大脑思考就草率的发问只能得到草率的回答，或者根本得不到任何答案便会石沉大海。其我特别不建议大家在QQ群或者微信群里向别人问问题，因为懂的人可能不屑于回答（觉得这样的问太low了，即使回答对了也体现不出自己的厉害），不懂的人即便回复你了也没有任何价值，反而有能会把你带偏了。现在这社会，大家的时间都很宝贵，没有哪个真正厉害的技术大牛是在QQ群和微群里活跃的，大牛的时间都很宝贵。那些成天在QQ群或者微信群里活跃的人，八成是闲的没事干的。

因此要想得到别人高质量的答复，必须拥有与之相匹配的高质量的问题才行，这样别人才会愿意帮你决。所以并不是什么问题都是值得向别人提问的。

我们在提问之前，一定要有自己的思考，优先尝试自己解决问题。下面是我提供的几个自我解决问题途径：

- 断点+日志相结合进行问题跟踪，深入源码探寻问题的本质并予以解决。
- 仔细通读作者编写的使用手册，不要拉下没一个细节（切忌想当然），试着自己找答案。
- 在FAQ里找答案。如果有开源地址的话，建议优先在Issue中查找是否有相关的问题，并借鉴其中解决方法。
- 在网上搜索相关问题（条件允许的话，优先使用google，百度太坑，搜到的大多千篇一律）
- 向你身边精于此道的朋友咨询。

如果经过以上5种方法你都没能解决问题，这个时候你再向别人进行提问，我相信你一定能够把问题决。因为带着思考向别人提问题，才更能够得到别人高质量的答复。如果可以的话，你可以直接把自想的几个解决方法阐述出来，这样别人可能会更愿意帮你解决，毕竟大家都喜欢做选择题，而不是论题。

清晰地描述问题

有这样一部分人，每次遇到问题需要向别人求助的时候，经常表现的是举足无措，慌张地描述了一大看上去和问题有关又或者无关的话，滔滔不绝口若悬河，问得被提问者一脸懵逼。

面对这样的问题难免会让别人头大。讲了一大堆却分不出主次和主要矛盾，就连问题是什么都没有搞很明白，别人怎么帮你解决？

因此我们在阐述问题的时候，需要注意以下几点：

- 问题描述要言简意赅，尽量控制在50字以内。
- 问题描述要条理清晰，把握主要矛盾。与问题无关或者相关性较低的话就不要说了。
- 建议问题中包括如下几部分的内容：
    - 问题描述
    - 使用版本
    - 如何重现
    - 期望的效果
    - 出错现象（截图或者视频）
    - 设备信息（环境）
    - 附加信息（可以是日志或者源码链接等）

问题解决及时反馈并表达谢意

现实生活中常常有这样一部分人，在得到别人帮助了之后，连一句问题是否被解决的答复或者感谢也有便消失得无影无踪，这会极大地打击被求助者的积极性。因为问题久拖未决会让人灰心，他们渴望到问题被解决，并从中得到帮助别人带来的满足感，这点非常重要。否则下次再有人向他提问题时，能就不太愿意帮忙了。

所以，在问题解决后，向所有帮助过你的人发个说明，让他们知道问题是怎样解决的，并再一次向他表示感谢。

微信公众号

更多资讯内容，欢迎扫描关注我的个人微信公众号！

原文链接：同样都是开发，为什么你不如别人？