# 那些会阻碍程序员成长的细节

罗马非一日建成，软件系统也不是一天能够写出来的，在经年累月的编码生活中，总会有那么些不经意的瞬间暴露出来，而这些不经意的外在表现日积月累，犹如水滴石穿，会产生巨大的力量反作用于程序员的成长。我简单列了几条，你来看一看，兴许就在身边实实在在发生过。

---

**拿到开发任务后，直接上手写代码。**缺少必要的沟通与设计，返工的机率极大。前后端数据的交互格式，功能潜在的关联点不清晰，接口调用方功能是否完备，存储结构的设计，复杂业务的流程设计等等，都需要事先沟通确定好，再动手写代码才能游刃有余，不然会走一步卡一步，进展缓慢，甚至倒退。

**在逻辑混乱的地方加入新东西，而不是去重构。**由于功能的新增或变更，需要在旧有的代码逻辑中添加新功能，本是一个很好的重构机会，但很多的做法时在原有的基础上填填补补，结果一片混乱。特别是在本已混乱的地方还要加入新代码逻辑，运行起来确实没有问题，但对自身代码质量的保证零意义。

**不愿意与别人分享技术成长。**教是学习最快的一条路，将自己所学传播分享给他人，并使他人能消化吸收，是对自己知识掌握一个最好的检验。同时在分享的过程中温故而知新，更加深对知识技能的掌握。如果你有教会徒弟饿死师傅的想法，会显得很落伍。互联网时代下，还有什么知识技能，是你独有而别人从来没有的？不如拿出来分享，大家共同成长。一个人走的快，一群人才走的远。

**遇到 BUG 首先否定是自己的问题。**这是一个普适性的问题，也是程序员遇到 BUG 时的第一反应。到底是不是别人的问题呢，往往是问题转了一圈又回到自己手里。耽误了大家的时间，同时降低的解决问题的效率。正解的姿势应该是立即检查自身，确定是不是自己的问题，是就立即改正，若不是，能找到问题所在最好，交由他人去处理，这是一种良好的习惯。

**缺乏验证条件时，开发的功能不经测试直接交付给测试人员。**一种是过于自信的表现，还有一种是懒惰的表现。有自信是好的，但如果能经过实际的场景来检验，双重保险，对自己对团队都是保证。懒惰就是不负责任的表现，有些功能确实测试起来很复杂，但为了保证功能的可用性，没有条件去创造条件也要完成，这也是一种态度。

**修复某 BUG 时，夹带其它问题。**一个未被测试人员发现的 BUG，自我发现后私自修复，并提交源码。这在测试阶段比较常见，但后期如果还出现这种问题，对产品/项目的稳定性是个极大的隐患，特别是生产环境。这是一个流程规范问题，也是一个职业素养问题。

先暂时写到这里吧，以上这些都不是大问题，但如果不注意，久而久之会演变成大问题。程序员是个特殊的物种，物种的进化依赖我们自身的知识结构、思维层次，更依赖于日常工作生活的三省其身，经常复盘总结回顾，才能发现问题，近而解决问题。

---

**领导安排什么就做什么，做完了就闲着，也不学习新业务新技能。**闲下来就刷刷微博朋友圈，看看新闻聊聊天，这是典型的打工者思维，上班工作，某种意义上也是自己创业的过程，在公司的体系下，利用对应的资源来达成

自己的目标，本质上与自己出来创业完成某个目标，路径是差不多的。在别人现有的环境里，不能以自己创业心态来待人接物，简单就是一个浪费，这是一个很好锻炼的机会，主动去承担更大的责任，更多的任务。诚然，做的多，出错的概率会更大，但试错的成本很低，成长的机会也更多，没有什么能比成长更好的啦！

**任务有交叉时，只关注自己的，不能从上下游全局统筹。**这其实是个合作意识的问题，需要双方或多方都比较爽才行，而不是只让自己爽，让别人很别扭。久而久之，自己会被孤立，这不是单次博弈，而是重复博弈，最优策略就是共赢。

**自己做的比较快，但旁边的兄弟任务似乎比较重，而自己又不施与援手。**个体技能的不同必然导致结果差异，但对于团队而言，需要的不仅仅是个体的成绩，而是团队的协作产生的高效率、高质量。这也是一个重复博弈的过程，难保自己有遇到困难的时候，你难道不希望有人搭把手共同解决问题？人同此心，心同此理，相互扶持，这是团队作战的魅力。

**不及时反馈工作进展。**做事最忌讳的就是虎头蛇尾，多数情况下会无疾而终，不了了之。对于被指派的任务最好的做法是及时上报风险、暴露困难，及时汇报进度，这样即便是达不到工作目标，整个过程也是有目共睹的。领导最怕的做法就是任务下发后，下面的也保证完成任务，但真到交活了，摆出一堆这样那样的理由，任务就是没有完成。当你没有把握给领导惊喜的时候，及时汇报工作进度，免得到最后只有惊吓，为时晚矣。

**伸手党。**到论坛里、社群里随便一看就能抓出来一堆，目的就是为了更快达成目标，但又能力不足或偷懒，而完全依赖于他人，最好是他把需求一

提，你把交付物一给就算完事。还有一种情况，出现问题后，不经深入分析，直接找别人来解决。长期伸手而不真正上手操练，业务能力自然下降，只能自尝苦果。有句话讲：你可以像只猪一样懒，却无法像只猪一样，懒得心安理得。这才是痛点。

**课余时间不充电。**下班后基本处于散养状态，工作基本吃老本，或者勉强能对付，工作上基本没什么大起色，不算差但也不算好。之前讲“你只有不停的奔跑，才能停留在原地！”，凸显出时代的变化之块，不是大鱼小鱼，而是快鱼吃慢鱼，特别在一些生活、工作节奏都比较慢的二三线城市。人与人之间的差距并不是在工作八小时内拉开的，起决定作用的而是工作之外，所谓工夫在诗外。技术更新迭代的速度远超任何一个时代，比我们优秀的人还在拼命，我们又有什么理由懒惰呢？

于细微处见真知，不断去尝试、调整、再尝试、坚持，学习是一辈子的事，不能停。

---

**不能主动推动事物前进。**主动做一件事跟被动接受去做事，心情都是不一样的，做事效率更是千差万别。主动的人有更多的成长机会，反之在被动中不断的响应别人的任务，这与处于那个层次高低无关。如果处于高层级却不能主动推进任务前进，相信在这个岗位上也不会呆太久。

**技术交流分享会仅仅是听听而已。**在一线城市这种机会特别多，如果你愿意，一天可以赶好几场。虽说是别人分享，如果没有好好利用，仅仅参加完就离开的话，效果几乎为零。可以事先熟悉讲师履历、分享的内容，带着

自己的疑问听，不懂处记录在册，答疑环节将疑问点抛出，多认识些分享会上的人，加入分享会微信群会后继续讨论等等，都能更好的发挥一次分享会的效能。不要让参加技术分享会流于形式。

**无法多场景切换角色。**刚入门时，醉心于开发还是有可能的，随着年限的增加，个人职责会发生改变，再有整块整块的时间去专心某一事务的机会不多，更多的是碎片化的时间切片，面对不同同事、客户、领导、合作伙伴等等，如果不能快速响应各种变化，只能用疲于奔命应付来形容了，工作的条理性更无从谈起。你需要快速切换角色，融入不同环境，能抽象，也能具化，能全局把握，也能细节深入，这样处理起问题才能得心应手，迅速成长。

**容易陷入细节。**或者说很纠结细节问题，相对的另一面会导致全局观缺失，偶尔还会钻牛角尖。过于追求细节完美的人，也就是完美主义者，事实时完美在现实生活中不可能存在。陷入细节而不自知轻则项目延期，重则错过市场窗口期。把握一个度很重要，需要以大局为重时，就需要牺牲局部的不完善。

**不插手开发任务外的其他事务。**临时任务在所难免，特别是创业团队，更是没有明确界线的岗位分工，需要时，直接顶上，目标就是把事情做成。仅关注开发任务那点事，显然不足以为更高的岗位做技能储备，如果力所能及又不占用过多时间的话，建议去锻炼一下。经历都算自己的，成长过，谁也拿不走。

**不注重关键节点的记录留存。**人员流动、业务变更对软件后续的维护都是挑战，仅靠脑袋记，显然不现实。很多朋友说，代码里都有啊，看代码就行了。如果必须拿代码才能说事的话，只能说你太不成熟，设计文档、流程

图、数据表格这些辅助理解的东西简直难道是摆设？特别是一些复杂的业务流程设计、复杂算法设计、疑难 BUG 解决方案等等。虽然敏捷开发不主张凡事以文档必备为准，但必要的文档留存很重要，形式可以不限。

不经意间，这些小细节慢慢写成了一个系列。俗语讲：魔鬼在细节中，小处做到极致，养成的好习惯，也是让自己未来的技术晋升之路更加坚实。

---

**不愿意跟领导走的近。**是不是有这样的体会：凡事有领导在的场合，气氛都比较凝重？整个人都放不开？其实这还是一个雇佣关系在作怪。员工与领导并不是处于一个合作互利互惠状态，也就是《联盟》一书中讲到的合作关系。如果还没看过的话，建议你去买来读一读。跟领导走的近有什么坏处吗？不能偷懒了是真的。在我看来，跟高管成为朋友是对职业生涯有益的事情。可以让领导更深入的了解你，可以站在比自己更高的维度去待人接物，可以接触到很多更高层次的人脉等等等等等，还有很多，待你去发掘。如果有机会，千万不要躲的远远的。

**缺乏理财意识。**IT 人薪资水平较一般行业而言还是比较可观的，但如果不注意保值、增值的话，月光是完全存在的，加上信用消费如此普及，卡里的数字会只减不增。再加上通胀，仅靠储蓄的话，实际上是资产贬值的过程。如此聪明的脑袋瓜，去接触一些股票、债券、基金、P2P、房产等标的的投资操作，还是很有必要的，你想实现财务自由吗？那就去提高你的睡后收入[被动收入]吧。

**频繁的变换行业。**信息化的触角已深入到各行各业，但行业与行业间的业务差别很大，而且有些行业的壁垒高、专业性很强，比如医药、财务等。

技术服务于业务，如果你的适应性很强，融会贯通的能力很强倒也无妨，但大部分情况下是刚接触一个还没深入就结束了，如果不能保持连贯，很可能到此就结束了。实际业务运行过程中的核心点需要一定的积累才能掌握，这就需要在行业里沉淀多年才能达到，比如专注金融行业，但里面也会涉及保险、证券、银行等几种业务场景，每一种又是繁杂多变，如果频繁变换从事的行业，基本上谈不上沉淀，随着工作年限的增长，自己的竞争力却没有提到提升，错失成为业务专家的能力。

**不停追逐新技术，由于不能应用于实际工作，导致很快被遗忘。**技术更新迭代的速度远非一个人穷一生之力所能学完的，所谓以有限的生命去掌握无限的技能仅仅是美好的事，但不现实。考虑到具体的技术成熟度、应用场景与自己的工作实践的匹配度，只能是那些切实应用于实践的技术栈才能更快更深的掌握住。具体到自己的职业规划中，更应当聚焦，找准一个点，基于这个点再外延，打造自己的技术生态圈。切忌把自己打造成一个掌握很多技术但无关度很高的瑞士军刀，其竞争力远不如一技之长。

**与他人沟通不畅时，不能本着解决问题的思想来解决问题。**“好，好，好，不说了，打住，我不跟你争了，就按你说的来，有问题你负责”。如果是本着解决问题的目标去解决问题，我们可以摆事实，讲论据，把各种方案存在的利弊得失放在明面上，然后大家再客观的就事论事，拿出一致的解决方案。你会时常碰到一些人在讨论问题时，最后的结果是一方不愿意再争论而妥协，但并非心服口服的妥协，这是直接撂挑子，而不是解决问题。

**跟踪问题不彻底，导致前功尽弃。**如果别人欠一大笔钱，想必你无论如何都会想尽一切办法，把这笔钱追要回来。很可惜的是，在日常的工作中，

我们跟踪一个任务的时候却远不如追账那么用心，跟着跟着然后就没然后了。可想而知，如果都这种状态的话，你的项目进度，你的销售目标，你的市场份额如何达成，试想有一天你成高层领导，你手下的兄弟都是这个状态跟踪问题，那你真的是得操碎了心也搞不定。

时隔几日，又啰啰嗦嗦的列了几条，希望里面都没有你的身影，如果碰巧有的话，也希望你摆正心态，积极的往前看，成长的道路没有一帆风顺。

---

**不依规范行事。**这个很常见，嘴上一套，行动时另一套，也就是知行不一。实际是知易行难，不自我监督，不自我约束，整个人都会变的懒散。“这个版本着急上线，先把功能实现再说，后面有时间再来重构优化”，是不是经常听到这样的套路，他们事后有重构吗？你自己事后有去看看哪些糟糕的代码吗？如果事出紧急事后完善也是允许的，就怕是个幌子，事后弃之九霄云外。

**想做却不迟迟不行动。**非得万事俱备了才行动，其实很多时永远没有完备的时候，先动起来再说，再行动的过程中肯定会遇到问题，去解决就是了。否则有的只是眼前的苟且，不会有诗和远方。“成功学”里讲，想成功与要成功是两回事，虽然是一字之差。

**不能虚心学习别人的长处。**都说文人相轻，其实 IT 界也有一条鄙视链，我们也总在有意无意见践行着这条鄙视链。即便是同一条链上的蚂蚱也存在相互不顺眼的时刻。但学习这东西，要虚心，不能说你看不惯他，就拒绝他身上一切事物，那怕是好的，这真的是跟自己过不去。你有看不惯的人吗？你发现他身上有你不具备的东西吗？

**会很多开发语言但不都精通。**不排除大牛做的到，一般人的话，你这么写简历，还真不会让人高看一眼，很容易给别人留下一个什么都会，基本上又

什么都不会的印象。不要追逐新技术，技术迭代的速度远比你学习的速度快的多，求多往往会不深入，止于皮毛，而一个技术的应用绝不是写个“hello world”就算的，应用到实际业务场景，解决实际的需求痛点才能发挥一项技术的本真。

**软件研发技能单一。**软件生命周期里除了编码实现，还有需求分析、系统设计、测试部署、发布上线、运维运营等等，只盯着代码那一块太狭隘，特别是工作时间长了之后，局限性会越来越明显。开发语言是工具，不用锤子用榔头一样敲东西。要提高自己的适应性，就得外延技能线，后面的职业路也会更宽广，而不是局限在技术一条线。追求全栈没有错，但要适度，不能盲目的为了简历上几行字，去涉猎那些看似有用，但实际并不会实际的技术。

**缺乏危机意识。**去年的热点话题事件就是中兴的一位 40 多岁的 IT 人自杀。雪球上针对这一问题爆发了“中年危机”的大讨论。中年危机是八零后绕不开的一个问题，张爱玲在《半生缘》里说，“中年以后的男人，时常会觉得孤独，因为他一睁开眼睛，周围都是要依靠他的人，却没有他可以依靠的人。”手机屏幕前的你考虑过这个问题吗？

---

**威胁性的处理问题。**比较常见的一种就是要挟式加薪，自恃岗位重要或者人员流动太大导致在岗人员变少等等，如果以此种理由去要求加薪有两种后果，一是领导暂时妥协，一旦找到替代者，直接卷铺盖走人，二是领导宁可人手阵痛，也不会助长这种气焰，结局还是卷铺盖走人。

**生活作息饮食不规律。**晚上睡不着，早上起不来。有事没事先来罐可乐，经常性的功能饮料替代白开水等等，不是说非得做个油腻猥琐男，手里捧着茶杯里面放上枸杞，但生活、身体还是自己的，一些不健康的饮食活动还是

尽量避免，偶尔满足一下但不能长期为之。IT 成高危职业早已不是什么新鲜，每年不猝死几个，仿佛这行业都不正常。

**喜欢收藏，而不是直接消化吸收。**这个更是个普适性问题，你现在就可以找找你的电脑硬盘里、微信收藏里、云盘里都藏了些啥？都看过吗？大部分的回答应该是先收藏等有时间再看，这跟松鼠无节制的收集松果是一样的。怎么破？先看了再说，如果没时间看的话，排个队列后期再看，如果感觉没时间看的话，索性也不要收藏了。

**不参与团队活动。**一个团队中部会发现一些比较安静的同事，特别是喜欢坐在一些角落里的同事，平时不出来活动，很容易忽略他的存在，更别提集体活动。这样非常不利于职场发展，金子发光也要让别人看到，更何况常有的情况是没有发光。同事间也需要维系一种职场情感，来形成职场工作的润滑剂，同时也是给自己积累人脉，日后难免有交集，在这个信息爆炸的时代，个体需要依赖于团队才能走的更远，如果自己仅作为一个团队的附庸，很难有所作为。

**不能自我激励。**当激情褪却，兴趣不再，如何再去敦促自己去完成一件原先兴冲冲的事呢？往往这个时候，事情就会搁置，不了了之，而成熟的人总会找到自我激励的方法。生活是自己的，没有谁应该在旁催着你促着你，如何不能自打鸡血，随着时间推移，说一事无成也不为过。

**忌惮与女生打交道。**都说 IT 男比较腼腆、比较闷、不修边幅、不善于与女生打交道等等，其实 IT 男不是闷更多的是闷骚。段子也是信手捻来，在网络上非常活跃。拿不出时间谈恋爱的人还是要拿时间来相亲，鉴于工作圈子的性质——女生比较少，导致很多 IT 男单身，所以更应该离开手机、电脑，

走向活生生的世界，走向单身妹子，该出手时就出手，不然一年中的每个节日都是光棍节。

---

**处理问题变成人身攻击。**不管软件开发，还是项目管理，最终落到实处都是人在执行。俗话讲有人的地方就有江湖，人与人难免会有摩擦，冲突是在所难免。但有时候处理冲突如果演变成人身攻击就变了味道，不但解决不了问题，还会树敌，更不利于工作的开展。就事论事，不牵扯个人情感因素在里面，事前可以吹胡子瞪眼，事后还是兄弟，一起喝酒吃串，这才是应有的策略。

**经历不少，但缺乏总结。**高中时期，为了所谓的读名著提升档次，确实是翻看了不少国外名家的名著。用“翻看”是确实就是如看小说般，有时候情节都没穿起来，更何谈什么写作手法，思想意境以及所映射的社会现象等。这就典型性的缺乏总结，流水账好记，难的是提炼出精华。如果经历很多，却不能从中萃取出经验，那也仅仅是多了一段经历而已，事后吹牛逼的谈资而已。

**好奇心不足。**举个栗子，面试时，发现不少朋友并没有做过注册的功能，我问他们，注册码那么丑，为什么还需要这么个东西在哪阻碍注册？注册码如何产生和验证的？注册码存储在哪里？虽然司空见惯，但真的是有朋友回答不清楚，越是熟悉的东西，越是熟视无睹。保持好奇心，才能探索到更多东西。

---

断断续续写了七篇不成体系的文章，着实有点对不住各位看官，毕竟可以分门别类去写，但成文的过程比较碎片化，难以成体系，只能有待后续再遇到主题时，重新梳理下。

其实每个人有一个开挂神器——学习。什么不会，马上可以去学。学以致用，可以立刻去解决问题。再加上思考，堪称相得益彰，相信看过印度神剧的朋友都知道，开挂的人生，无往而不利。

难能可贵的我们能发现自身不足，并从点滴改善，慢慢去精进去修炼。有朋友问，他已经发现了问题所在，但就是迟迟不愿意行动怎么办？我的回答是：那说明你还没意识到问题的严重性，当你认识到了，就不会跑过来问我怎么办，而是已经悄悄的去做了！