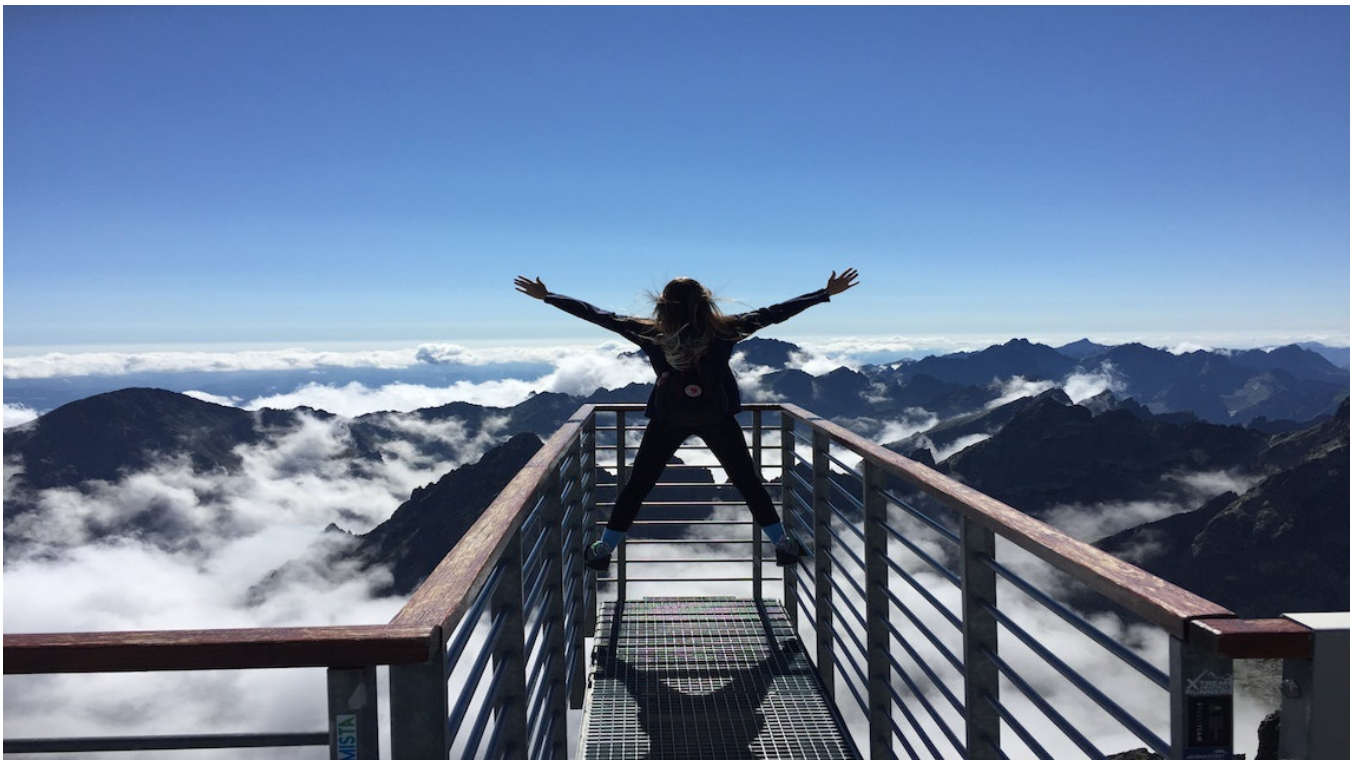
27 | 试试：一种“坏”习惯

2018-10-03 胡峰



曾经，我碰到一些程序员问我：“我以前是做安卓的，现在想试着学下后端服务开发，你觉得怎样？”我一下子就卡住了，不知该如何回答才好。原因是：学习本是个好事，但前面加个“试着”似乎感觉就不太好了。

好的出发点

“试一试”的初衷本来就该是好的，它表达了一种好奇心，以及尝试走出舒适区的勇气。

程序员这个职业，会带来一些职业习惯。比如，可能会经常性地去尝试一些新东西，然后看看它是否如预期般那样被应用或实现。

这里，我就拿程序员“调试程序”这项日常工作来举例。调试，就是这样一种需要不断去试的过程。

还记得我在前面《炫技与克制》一文中讲了我早年刚开始工作时的那个小故事吗？那时我带着炫技的心态应用了刚刚接触的 Java 线程编程通信来实现一个客户端小程序。结果后来程序出了 Bug，而我不断修改，Bug 从这里消失，又从那里冒出来，让那时的我产生了巨大的挫败感。

当时，我花了很长时间一直在“抓”这个 Bug，用的方法就是调试技术。但因为这是一个机率性出现的 Bug，一步步调试反而从来没出现过，但真正运行起来又总是偶然出现，实在让人抓狂。在这样的单步调试中，我就是怀着一种期望凑巧能碰到的心态，做了很多无用功，最后也没能解决真正的问题。

这个案例虽然已经过去了十几年，但还是给我留下了深刻的印象，久久不能忘怀。我把它分享出来，就是感觉想必这条路上曾经的我不会是特例。

表面上看，是当时那种炫技的心态致使我选择了不恰当的实现方案，也最终导致出现了对于那时的我来讲很难解决的 Bug。但其实这里真正的症结是：我对于线程间通信的知识出现了认知性盲点，这属于 “我以为自己知道，其实不知道” 的问题。

我习惯性地用调试去找 Bug，这就是一种 “试一试” 的方法，出发点（找到 Bug）是好的，过程也是很艰辛的，但最终结果却是无功而返。即便用这样的方法最终找到了 Bug，也有一定的运气因素，并不具备可重复性。

当时，我正在读一本有关线程编程的书，后来读到某个部分时，关于这个问题的根源我突然就恍然大悟了，因为这个部分正好弥补了“我以为自己知道，实际却不知道”的认知盲点。我习惯性的调试方法，虽然有一个好的出发点，但问题是，我不知道我在调试什么。也许是想通过调试证明程序逻辑本不该出错的，或是通过调试发现其他的疏漏，但在这样的盲目调试中最终也没能定义清楚我调试的终点到底是怎样的。

那时的我就是一个刚进入编程领域的小白，喜欢调试，然后在看上去很复杂的调试界面忙忙碌碌，感觉很专业，但最终收获的仅仅是对调试器的熟悉程度。而且一不留神，就自觉不自觉地养成了这种“试一试”的“坏”习惯。

模糊的终点

这里，“试一试”的“坏”习惯的“坏”字之所以加上双引号，就在于它的出发点本是好的，但如果终点是模糊的，那就“坏”了。

近些年来，就出现过几轮的技术热，比如，刚进入移动互联网时代就大热、但如今已经回归常温的移动开发，曾经大热现已降温的云计算与大数据，以及还在热度中的人工智能、机器学习和区块链等。面对这些技术热，很多人都跃跃欲学之、试之。可能你也不例外。那么，到底为什么你会想去尝试一种新技术？是你仔细思考后的主动选择，还是说或多或少又被技术潮流所裹挟？

好些年前，移动开发还在升温阶段时，我也不可避免地被这样一种潮流所裹挟过。我开始看一些关于 iOS 开发的书，从语言到工具。其实，尝试学习一种新技术并不是坏事，即使是被技术潮流所裹挟，但问题出在，这次尝试的终点在哪里？

我是想转型成为一名移动开发工程师吗？还是说我有一个想法，需要开发一个 App 来达成？抑或我仅仅是想学习并了解下移动开发是怎么回事，从而进一步提升下技术的广度理解与视野？

然而以上皆不是，我当时的尝试完全没想清楚终点在哪儿。后来热度下来了，其他工作任务也多了，也就慢慢遗忘了。回过头来看，这只是浪费了一些时间和精力罢了。

几年后，人工智能与机器学习又热了起来，我又开始尝试学习起来，但较上次不同的是，这次我

把尝试的终点定义得很清楚。我不是想转型成为一名机器学习领域的算法工程师，也不是因为它很热就“随波逐流”地被潮流裹挟，我这次尝试的终点就是想搞清楚关于人工智能与机器学习的三件事：

- 它的原理与应用场景；
- 它的前世今生；
- 它如今已抵达的边界。

搞清楚这三件事，虽不会让我成为机器学习的专家，但会提升我对于这个热门技术的判断力。因为，现实中我需要判断一些真实的业务场景该如何结合这样的技术，这就需要了解它们的应用场景和一些原理。

另外，一门新技术很少是凭空冒出来的，了解它们的前世今生，会更有效地知道，哪些方面已经有了成熟的方案，哪些地方还在青涩的探索期。再结合它当前的边界，就知道如何定义清楚需要，形成合理的技术方案，而不会产生过度的妄想。

试一试，需要有更清晰的终点。关于终点，你也可以从下面一些方面来考虑：

1. **验证猜想**。这个部分程序员就很熟悉了，因为编程中的调试其实最重要的目的就是验证猜想。引入一种新技术或框架，验证 API 的调用结果或运行输出是否如你所想，即使最终否决了，那你也获得了判断的依据与知识。

2. **收获结果**。定义清楚你尝试的这件事，到底能收获怎样具体的结果。比如：考试，尝试的收获就是要通过。

3. **体验过程**。有时候结果并不确定，比如，创业的结果未必就一定是成功，那么这样的尝试难道就没有意义了吗？有的，因为创业的超低成功率，所以，体验过程恐怕多于收获最终结果。

4. **理解现实**。你尝试一个新东西或学习一个新知识，有时未必真是为了将来有朝一日能用上它，而主要是为了完善你的知识与认知体系，然后再去理解现实为什么是这样的。

现实的路径

“试一试” 的路径是有限的，毕竟终究离不开现实的约束。

有时候，你因为现实工作需要，可能需要不停地在各种技术栈上切换。而很多技术可能过了那段时间，就再也用不上了，这样的技术尝试难免会让人感觉可惜。但通过我前面列出的关于 “终点” 的方面，再来分析下这个现实场景。

首先，你得面对现实，这样的技术尝试在现实中太多太多了，有时就是没得选择。当年，我也因为工作原因，从客户端桌面编程的 VB、PB、Delphi 到 Web 编程的 JS 语言和一堆相关框架，再到后端编程的 C 和 Java，而如今很多当年学习的技能早已过时了。但这样的技术切换尝试，

从“收获结果”的维度看还是解决了当时的问题，满足了需要，获得了结果。

其次，如果觉得仅仅一次性收获的结果，不值得你投入的时间和精力，那就可以从“理解现实”的角度去挖掘。这些知识，从学以致用的角度很快就过时了，但它们并不是完全孤立的，事实上计算机程序体系内的很多知识都不是完全孤立的，它们都有相互的联系与连接点。

从理解的角度，这类技术切换的尝试事实上扩大了你的知识边界，尝试的也许是孤点，但你可以进一步找到它们的连接处，形成体系。因为很多现实的原因，每个人的起点和路径都不会一样，但我们都是从某一点开始去慢慢摸索、尝试，最终走出一个属于自己的体系来的。

最后，当你有了自己的体系，也可能有了更多的尝试选择权，就可以体系为中心，去有选择地尝试对你更有意义或价值的事了。

总结来说：

试一试，是走出舒适区的一次行动，这本是一个好的出发点，但若只有一个模糊的终点，那么它带来的更可能就是无谓的浪费。

试一试，不仅要有一个好的出发点，还需要一个清晰的终点，在这个终点你可能：验证猜想、收获结果、体验过程、理解现实。而在起点和终点之间，你需要选择一条更现实的路径，通过不断地尝试，走出自己的体系。

试一试，本该是个好习惯，可别把它用坏了。

想必很多人都有过盲目尝试的经历，欢迎留言分享你的经历及反思。
